Numerically Safe Gaussian Elimination with No Pivoting *

Victor Y. Pan[1,2],[a] and Liang Zhao[2],[b]

[1] Department of Mathematics and Computer Science
Lehman College of the City University of New York
Bronx, NY 10468 USA
[2] Ph.D. Programs in Mathematics and Computer Science
The Graduate Center of the City University of New York
New York, NY 10036 USA
[a] victor.pan@lehman.cuny.edu
http://comet.lehman.cuny.edu/vpan/
[b] lzhao1@gc.cuny.edu

Abstract

Gaussian elimination with no pivoting and *block Gaussian elimination* are attractive alternatives to the customary but communication intensive *Gaussian elimination with partial pivoting*[1] provided that the computations proceed *safely* and *numerically safely*, that is, run into neither division by 0 nor numerical problems. Empirically, safety and numerical safety of GENP have been consistently observed in a number of papers where an input matrix was pre-processed with various structured multipliers chosen ad hoc. Our present paper provides missing formal support for this empirical observation and explains why it was elusive so far. Namely we prove that GENP is numerically unsafe for a specific class of input matrices in spite of its pre-processing with some well-known and well-tested structured multipliers, but we also prove that GENP and BGE are safe and numerically safe for the average input matrix pre-processed with any nonsingular and well-conditioned multiplier. This should embolden search for sparse and structured multipliers, and we list and test some new classes of them. We also seek randomized pre-processing that universally (that is, for all input matrices) supports (i) safe GENP and BGE with probability 1 and/or (ii) numerically safe GENP and BGE with a probability close to 1. We achieve goal (i) with a Gaussian structured multiplier and goal (ii) with a Gaussian unstructured multiplier and alternatively with Gaussian structured augmentation. We consistently confirm all these formal results with our tests of GENP for benchmark inputs. We have extended our approach to other fundamental matrix computations and keep working on further extensions.

2000 Math. Subject Classification: 15A06, 15A52, 15A12, 65F05, 65F35

Keywords: Gaussian elimination; Pivoting; Block Gaussian elimination; Preconditioning; Random matrix algorithms; Linear systems of equations

*Some results of this paper have been presented at the 17th Annual Conference on Computer Algebra in Scientific Computing (CASC'2014), September 10–14, 2015, Aachen, Germany (cf. [PZ15]).

[1] Hereafter we use the acronyms *GENP*, *BGE*, and *GEPP*.

1 Introduction

1.1 Gaussian elimination with pivoting

The history of Gaussian elimination can be traced back some 2000 years [G11]. Its modern version, GEPP (with partial pivoting), is performed routinely, millions times per day around the world, being a cornerstone for computations in linear algebra [DDF14]. For an $n \times n$ matrix, elimination involves about $\frac{2}{3}n^3$ flops, and $(n-1)n/2$ comparison are required for partial pivoting, that is, row interchange.[2] Clearly pivoting contributes a substantial share to the overall computational cost if n is small, but even for larger n communication intensive pivoting takes quite a heavy toll in modern computer environment. It interrupts the stream of arithmetic operations with foreign operations of comparison, involves book-keeping, compromises data locality, impedes parallelization of the computations, and increases communication overhead and data dependence. According to [BDHT13], "pivoting can represent more than 40% of the global factorization time for small matrices, and although the overhead decreases with the size of the matrix, it still represents 17% for a matrix of size 10,000". Because of the heavy use of GEPP, even its limited improvement is valuable.

1.2 Random and nonrandom multipliers. Safety and numerical safety

Gaussian elimination with no pivoting (GENP) is an attractive alternative to GEPP[3], but for some inputs can be *unsafe* or *numerically unsafe*, that is, can run into a division by 0 or numerical problems, respectively. Empirically, GENP is quite consistently safe and numerically safe [PQZ13], [BDHT13], [PQY15], [DDF14] if the input matrix is pre-processed with various structured multipliers chosen ad hoc (e.g., with random circulant or SRTF multipliers),[4] but formal support for this empirical observation turned out to be elusive.

We explain why it is elusive by exhibiting a class of input matrices for which GENP with a random circulant or SRFT multiplier is always numerically unsafe (see Section 7.2), but we also explain why pre-processing with such multipliers usually works – we prove that *GENP is both safe and numerically safe for the average input matrix pre-processed with any nonsingular and well-conditioned multiplier* provided that the average is defined under the Gaussian probability distribution (see Section 4.2), which is a customary assumption in view of the Central Limit Theorem.

In Section 6 we list and in Section 9 successfully test some promising classes of such sparse and structured multipliers, and plan to work further in this direction. For inputs to our tests we used *benchmark matrices* from [BDHT13] or applied the *customary recipes* of [H02, Section 28.3].

1.3 Universal pre-processing

In addition to our results on pre-processing GENP with a fixed multiplier for the average input matrix, we prove that pre- as well as post-multiplication by a Gaussian multiplier[5] are *universally safe*, that is, safe with probability 1 for any fixed nonsingular input matrix, and is *universally numerically safe*, that is, likely to be numerically safe for any nonsingular and well-conditioned input matrix (see Section 4.2). Actually we prove that it is numerically safe with a higher probability if the input matrix is both pre- and post-multiplied by Gaussian matrices (see Remark 4.4).

In computations with infinite precision and with no rounding errors (e.g., in Computer Algebra) one needs just safe (rather than numerically safe) GENP, and we prove universal safety of GENP even in the case of pre- or post-multiplication by a random Gaussian circulant matrix (see Section 7). This result immediately enables 4-fold acceleration of the classical, extensively cited, and highly popular two-sided pre-processing of [KS91].

[2] Here and hereafter "flop" stands for "floating point arithmetic operation".

[3] Another alternative is symmetrization, but it has deficiencies for both numerical and symbolic computations: it squares the condition number of the input matrix and does not work over finite fields.

[4] SRFT is the acronym for Semisample Random Fourier Transform

[5] Here and hereafter "Gaussian matrices" and "Gaussian pre-processing" stand for "standard Gaussian random matrices" and "standard Gaussian random pre-processing", respectively.

We cannot prove universal numerical safety of GENP pre-processed with SRFT or any other random structured multipliers, but we prove that GENP pre-processed with *SRFT augmentation* or *SRFT additive preprocessing* is universally safe with probability 1 and is likely to be universally numerically safe (see Section 8.2).

Compared to [PQZ13], [BDHT13], [PQY15], and [DDF14], our present tests cover pre-processing also with multipliers from new classes as well as by means of augmentation; most important, now our test results are in very good accordance with our new formal results.

1.4 Block Gaussian elimination

Block matrix algorithms are highly important resource for enhancing the efficiency of matrix computations [GL13], but *block Gaussian elimination* (BGE), including, e.g., the MBA celebrated superfast algorithm for solving structured linear systems of equations, is prone to numerical stability problems (cf. [B85], [P01, Chapter 5]). In Section 3, however, we readily extend our results for GENP to BGE.[6] In particular a proper random structured pre-processing is likely to make BGE safe and numerically safe. This enables us to resurrect the MBA algorithm for numerical computations (see our Remark 5.6). In Remark 4.3 we accelerate our pre-processing for BGE by performing it recursively for leading block submatrices rather than once for the whole matrix.

1.5 Organization of the paper

We organize our presentation as follows. In the next section we cover basic definitions and some preliminary results. In Section 3 we define BGE and show that, for any input, safety as well as numerical safety of GENP imply the same properties also for BGE. In Section 4 we prove our basic theorems about safety and numerical safety of GENP (and consequently BGE) with multiplicative pre-processing and at the end cover some techniques for enhancing the power and efficiency of pre-processing. In Section 5 we recall the definitions and some basic properties of the matrices of Discrete Fourier transform and circulant and factor-circulant (f-circulant) matrices. In Section 6 we cover some families of efficient multipliers. In Section 7 we prove that randomized circulant pre-processing for GENP and BGE is universally safe (the contribution of the second author) but is numerically unsafe for some specific input class of matrices. In Section 8 we prove universal safety of augmentation and additive pre-processing with Gaussian as well as SRFT pre-processors. Section 9 (also the contribution of the second author) covers our numerical experiments with benchmark inputs and inputs generated according to [H02, Section 18.3]. In Section 10 we briefly recall our progress and then comment on its extension to low-rank approximation and other fundamental matrix computations. A number of our results for Gaussian pre-processing can be extended to pre-processing under the uniform probability distribution on a finite set (cf. Theorem 2.1).

2 Basic definitions and preliminary results

Hereafter "likely" and "unlikely" mean "with a probability close to 1" and, respectively, "to 0". We call an $m \times n$ matrix *Gaussian* and denote it $G_{m,n}$ if all its entries are i.i.d. standard Gaussian variables. $\mathcal{G}^{m\times n}$, $\mathbb{R}^{m\times n}$, and $\mathbb{C}^{m\times n}$ denote the classes of $m\times n$ Gaussian, real and complex matrices, respectively. In order to simplify our presentation, we assume dealing with real matrices, except for the matrices of discrete Fourier transform of Section 5.1 and f-circulant matrices of Section 5.2 used in Sections 6 and 9, but we can readily extend our study to the complex case.

2.1 General matrices: basic definitions

In this subsection we recall some basic definitions for general matrix computations (cf. [GL13]).

1. I_g is a $g \times g$ identity matrix. $O_{k,l}$ is the $k \times l$ matrix filled with zeros.

[6] By combining our dual approach of Section 4.3 with the study in [YC97] of the growth factor in PLUP' factorization of the input matrix, we can deduce the results similar to our present ones for GENP but not for BGE.

2. $(B_1 \mid B_2 \mid \cdots \mid B_k)$ is a block vector of length k, and $\mathrm{diag}(B_1, B_2, \ldots, B_k)$ is a $k \times k$ block diagonal matrix, in both cases with blocks $B_1, B_2, \ldots, B_k$.

3. $W_{k,l}$ denotes the $k \times l$ leading (that is, northwestern) block of an $m \times n$ matrix W for $k \le m$ and $l \le n$. A matrix is *strongly nonsingular* if all its square leading blocks are nonsingular.

4. $\mathcal{R}(W)$, W^T and W^H denote its range (that is, column span), transpose and Hermitian transpose, respectively. $W^H = W^T$ for a real matrix W.

5. An $m \times n$ matrix W is called *unitary* (in the real case also and preferably *orthogonal*) if $W^H W = I_n$ or $W W^H = I_m$.

6. $Q(W)$ denotes the matrix obtained by means of column orthogonalization of a matrix W, followed by the deletion of the columns filled with zeros (cf. [GL13, Theorem 5.2.3]).

7. $||W||$ and $||W||_F$ denote its spectral and Frobenius norms, respectively.

8. $S_W \Sigma_W T_W^T$ is its full Singular Value Decomposition (or full SVD) of an $m \times n$matrix W where S_W and T_W are square unitary matrices and $\Sigma_W = \mathrm{diag}(\mathrm{diag}(\sigma_j(W))_{j=1}^{\rho}, O_{m-\rho,n-\rho})$, for $\rho = \mathrm{rank}(W)$, is the $m \times n$ diagonal matrix of the singular values of the matrix W,
$$\sigma_1(W)) \ge \sigma_2(W)) \ge \cdots \ge \sigma_\rho(W)) > 0,\ \sigma_j(W)) = 0 \text{ for } j > \rho.$$

9. $W = S_{W,\rho} \Sigma_{W,\rho} T_{W,\rho}^T$ is its compact SVD, where $\Sigma_{W,\rho} = \mathrm{diag}(\sigma_j(W))_{j=1}^{\rho}$ is the $\rho \times \rho$ leading submatrix of Σ_W and the matrices $S_{W,\rho}$ and $T_{W,\rho}$ are formed by the first ρ columns of the matrices S_W and T_W, respectively.

10. $W^+ = T_{W,\rho} \Sigma_{W,\rho}^{-1} S_{W,\rho}^T$ is its Moore–Penrose pseudo inverse.
$(W^+)^+ = W$, $W W^+ = I_m$ if $\mathrm{rank}(W) = m$, $W^+ W = I_n$ if $\mathrm{rank}(W) = n$, and $W^+ = W^{-1}$ for a nonsingular matrix W.

11. $||W|| = \sigma_1(W)$ and $||W||_F = (\sum_{j=1}^{\rho} \sigma_j^2(W))^{1/2} = (\mathrm{Trace}\ (W^H W))^{1/2}$ denote its spectral and Frobenius norms, respectively.
$||VW|| \le ||V||\ ||W||$ and $||VW||_F \le ||V||_F ||W||_F$, for any matrix product VW.
$||U|| = ||U^+|| = 1$, $||UW|| = ||W||$ and $||WU|| = ||W||$ if the matrix U is unitary.

12. $\sigma_\rho(W) = 1/||W^+||$. $\kappa(W) = ||W||\ ||W^+|| = \sigma_1(W)/\sigma_\rho(W) \ge 1$ is the condition number of W.

13. The ξ-rank of a matrix, for a positive ξ, is the minimum rank of its approximations within the norm bound ξ. The *numerical rank* of a matrix is its ξ-rank for ξ small in context.

14. A matrix W is *ill-conditioned* if its condition number is large in context or equivalently if its rank exceeds its numerical rank. The matrix is *well-conditioned* if its condition number is reasonably bounded. The ratio of the output and input error norms of Gaussian elimination is roughly the condition number of an input matrix (cf. [GL13]).

2.2 Random matrices: definitions, some basic properties

We use the acronym " *i.i.d.*" for "independent identically distributed", keep referring to standard Gaussian random variables just as *Gaussian*, and call random variables *uniform* over a fixed finite set if their values are sampled from this set under the uniform probability distribution on it.

The matrix is *Gaussian* if all its entries are i.i.d. Gaussian variables.

Theorem 2.1. *Suppose that A is a nonsingular $n \times n$ matrix and H is an $n \times n$ matrix whose entries are linear combinations of finitely many i.i.d. random variables and let* $\det((AH)_{l,l})$ *vanish identically in them for neither of the integers l, $l \le n$. (i) If the variables are uniform over a set $\mathcal{S}$ of cardinality $|\mathcal{S}|$, then the matrix $(AH)_{l,l}$ is singular with a probability at most $l/|\mathcal{S}|$, for any l, and so the matrix AH is strongly nonsingular with a probability at least $1 - 0.5(n-1)n/|\mathcal{S}|$. (ii) If these i.i.d. variables are Gaussian, then the matrix AH is strongly nonsingular with probability 1.*

Proof. Part (i) of the theorem follows from a celebrated lemma of [DL78]. Derivation is specified, e.g., in [PW08]. Claim (ii) follows because the equation $\det((AH)_{l,l})$ for any integer l in the range from 1 to n defines an algebraic variety of a lower dimension in the linear space of the input variables (cf. [BV88, Proposition 1]). □

Lemma 2.1. *(*Orthogonal invariance of a Gaussian matrix.*) Suppose that k, m, and n are three positive integers, $k \le \max\{m, n\}$, G is an $m \times n$ Gaussian matrix, and S and T are $k \times m$ and $n \times k$ orthogonal (or unitary) matrices, respectively. Then SG and GT are Gaussian matrices.*

2.3 Norm of a Gaussian matrix and of its pseudo inverse

Next we recall some estimates for the norms of a Gaussian matrix and of its pseudo inverse. For simplicity we assume that we deal with real matrices, but similar estimates in the case of complex matrices can be found in [CD05] and [ES05]. Hereafter we write $\nu_{m,n} = ||G||$ and $\nu^+_{m,n} = ||G^+||$ for a Gaussian $m \times n$ matrix G, and write $\mathbb{E}(v)$ for the expected value of a random variable v.

Theorem 2.2. *(Cf. [DS01, Theorem II.7].) Suppose that m and n are positive integers, $h = \max\{m, n\}$, $t \ge 0$, and $z \ge 2\sqrt{h}$. Then (i)* Probability$\{\nu_{m,n} > t + \sqrt{m} + \sqrt{n}\} \le \exp(-t^2/2)$ *and*
(ii) $\mathbb{E}(\nu_{m,n}) \le \sqrt{m} + \sqrt{n}$.

Theorem 2.3. *Let $\Gamma(x) = \int_0^\infty \exp(-t)t^{x-1}dt$ denote the Gamma function and let $x > 0$. Then*
(i) Probability $\{\nu^+_{m,n} \ge m/x^2\} < \frac{x^{m-n+1}}{\Gamma(m-n+2)}$ *for $m \ge n \ge 2$,*
(ii) Probability $\{\nu^+_{n,n} \ge x\} \le 2.35\sqrt{n}/x$ *for $n \ge 2$,*
(iii) $\mathbb{E}((\nu^+_{F,m,n})^2) = m/|m - n - 1|$, *provided that $m > n + 1 > 2$, and*
(iv) $\mathbb{E}(\nu^+_{m,n}) \le e\sqrt{m}/|m - n|$, *for $e = 2.7182818\ldots$, provided that $m \ne n$.*

Proof. See [CD05, Proof of Lemma 4.1] for claim (i), [SST06, Theorem 3.3] for claim (ii), and [HMT11, Proposition 10.2] for claims (iii) and (iv). □

Probabilistic upper bounds on $\nu^+_{m,n}$ of Theorem 2.3 are reasonable already for square matrices, for which $m = n$, but become much stronger as the difference $|m - n|$ grows above 2.

Theorems 2.2 and 2.3 combined imply that an $m \times n$ Gaussian matrix is very well-conditioned if the integer $m - n$ is large or even moderately large, and still can be considered well-conditioned if the integer $|m - n|$ is small or even vanishes (possibly with some grain of salt in the latter case). These properties are immediately extended to all submatrices because they are also Gaussian.

3 Recursive Factorization of a Matrix. BGE and GENP

In this section we specify recursive factorization of a strongly nonsingular matrix, which we will use as a basis for our simultaneous study of safety and numerical safety of GENP and BGE.

For a nonsingular 2×2 block matrix $A = \begin{pmatrix} B & C \\ D & E \end{pmatrix}$ of size $n \times n$ with nonsingular $k \times k$ *pivot block* $B = A_{k,k}$, define $S = S(A_{k,k}, A) = E - DB^{-1}C$, the *Schur complement* of $A_{k,k}$ in A, and the block factorizations,

$$A = \begin{pmatrix} I_k & O_{k,r} \\ DB^{-1} & I_r \end{pmatrix} \begin{pmatrix} B & O_{k,r} \\ O_{r,k} & S \end{pmatrix} \begin{pmatrix} I_k & B^{-1}C \\ O_{k,r} & I_r \end{pmatrix}, \tag{3.1}$$

$$A^{-1} = \begin{pmatrix} I_k & -B^{-1}C \\ O_{k,r} & I_r \end{pmatrix} \begin{pmatrix} B^{-1} & O_{k,r} \\ O_{r,k} & S^{-1} \end{pmatrix} \begin{pmatrix} I_k & O_{k,r} \\ -DB^{-1} & I_r \end{pmatrix}. \tag{3.2}$$

These factorizations represent Gauss-Jordan elimination applied to a 2×2 block matrix.

We readily verify that S^{-1} is the $(n - k) \times (n - k)$ trailing (that is, southeastern) block of the inverse matrix A^{-1}, and so the Schur complement S is nonsingular since the matrix A is nonsingular.

Factorization (3.2) reduces the inversion of the matrix A to the inversion of the leading block B and its Schur complement S, and we can recursively reduce the inversion task to the case of

the leading blocks and Schur complements of decreasing sizes as long as the leading blocks are nonsingular. After sufficiently many recursive steps of this process of BGE, we only need to invert matrices of small sizes, and then we can stop the process and apply a selected black box inversion algorithm, e.g., based on orthogonalization. If we limit the number of recursive steps, we arrive at BGE dealing with large blocks and can use the benefits of block matrix algorithms, but if we keep recursive partitioning, then BGE eventually turn into GENP.

Namely, in $\lceil \log_2(n) \rceil$ recursive steps all pivot blocks and all other matrices involved into the resulting factorization turn into scalars, all matrix multiplications and inversions turn into scalar multiplications and divisions, and we arrive at a *complete recursive factorization* of the matrix A. If $k = 1$ at all recursive steps, then the complete recursive factorization (3.2) defines GENP.

Moreover, any complete recursive factorizations turns into GENP up to the order in which we consider its steps. This follows because at most $n-1$ distinct Schur complements $S = S(A_{k,k}, A)$, for $k = 1, \dots, n-1$, are involved in all recursive block factorization processes for $n \times n$ matrices A, and so we arrive at the same Schur complement in a fixed position via GENP and via any other recursive block factorization (3.1). Hence we can interpret factorization step (3.1) as the block elimination of the first k columns of the matrix A, which produces the matrix $S = S(A_{k,k}, A)$. If the dimensions $d_1, \dots, d_r$ and $\bar{d}_1, \dots, \bar{d}_{\bar{r}}$ of the pivot blocks in two block elimination processes sum to the same integer k, that is, if $k = d_1 + \cdots + d_r = \bar{d}_1 + \cdots + \bar{d}_{\bar{r}}$, then both processes produce the same Schur complement $S = S(A_{k,k}, A)$. The following results extend this observation.

Theorem 3.1. *In the recursive block factorization process based on (3.1), the diagonal block and its Schur complement in every block diagonal factor is either a leading block of the input matrix A or the Schur complement $S(A_{h,h}, A_{k,k})$ for some integers h and k such that $0 < h < k \le n$ and $S(A_{h,h}, A_{k,k}) = (S(A_{h,h}, A))_{h,h}$.*

Corollary 3.1. *The complete recursive block factorization process based on equation (3.1) can be computed by involving no singular pivot blocks (and, in particular, no pivot elements vanish) if and only if the input matrix A is strongly nonsingular.*

Proof. Combine Theorem 3.1 with the equation $\det A = (\det B) \det S$, implied by (3.1). □

4 Multiplicative Pre-processing for GENP and BGE

4.1 Definition and criteria of safety and numerical safety

In this section, A denotes a nonsingular $n \times n$ matrix.

Suppose that the vector $\mathbf{y} = A\mathbf{b}$ satisfies pre-processed linear systems $AH\mathbf{y} = \mathbf{b}$ and $FAH\mathbf{y} = F\mathbf{b}$. Then the vector $\mathbf{x} = H\mathbf{y}$ for $\mathbf{y} = A^{-1}\mathbf{b}$ satisfies both linear systems $A\mathbf{x} = \mathbf{b}$ and $FA\mathbf{x} = F\mathbf{b}$. We are going to study such pre-processing $A \to AH$ for GENP and BGE with random and fixed post-multipliers H. Our analysis is immediately extended to the pre-processing maps $A \to FA$, $A \to FAH$, and $A \to FAF^T$.

We call GENP and BGE *safe* if they proceed to the end with no divisions by 0.

Corollary 3.1 implies the following result for computations in any field (cf. Remark 4.1).

Theorem 4.1. *GENP is safe if and only if the input matrix is strongly nonsingular.*

Next assume that GENP and BGE are performed numerically, with rounding to a fixed precision, e.g., the IEEE standard double precision. Then extend the concept of safe GENP and BGE to *numerically safe GENP and BGE* by requiring that the input matrix be strongly nonsingular and *strongly well-conditioned*, that is, that the matrix itself and all its square leading blocks be nonsingular and well-conditioned. Any inversion algorithm for a nonsingular matrix is highly sensitive to the input and rounding errors if the matrix is ill-conditioned [GL13]. GENP explicitly or implicitly involves the inverses of all its square leading blocks, and we arrive at the following *Criterion of Numerical Safety of GENP* implied by [PQZ13, Theorem 5.1]:

GENP applied to a strongly nonsingular matrix is highly sensitive to the input and rounding errors if and only if some of the square leading blocks are ill-conditioned.

Let us restate this criterion in the form similar to Theorem 4.1.

Theorem 4.2. *GENP is safe and numerically safe if and only if the input matrix is strongly nonsingular and strongly well-conditioned.*

Remark 4.1. *BGE is safe if so does GENP. Likewise BGE is safe numerically if so does GENP. Thus our proofs of safety and numerical safety of GENP apply to BGE. The converse is not true: GENP fails (resp. fails numerically) if any square leading block of the input matrix is singular (resp. ill-conditioned), but BGE may by-pass this block and be safe (resp. numerically safe).*

4.2 GENP with Gaussian pre-processing

Next we prove that GENP with Gaussian pre-processing is safe with probability 1 for any nonsingular input matrix and is likely to be numerically safe if this matrix is also well-conditioned.

Theorem 4.3. *Assume that we are given a nonsingular and well-conditioned $n \times n$ matrix A and a pair of $n \times n$ Gaussian matrices F and H and let $\nu^+_{k,k}$, $\nu^+_{k,n}$, and $\nu^+_{n,k}$ denote random variables of Section 2.3. Then*

(i) the matrices FA, AH, and FAH are strongly nonsingular with a probability 1,

(ii) $||((AH)_{k,k})^+|| \le \nu^+_{k,k}||A^+_{k,n}|| \le \nu^+_{k,k}||A^+||$, $||((FA)_{k,k})^+|| \le \nu^+_{k,k}||A^+_{n,k}|| \le \nu^+_{k,k}||A^+||$, and

(iii) $||((FAH)_{k,k})^+|| \le \nu^+_{k,n}\nu^+_{n,k} \min\{||A^+_{k,n}||, ||A^+_{n,k}||\} \le \nu^+_{k,n}\nu^+_{n,k}||A^+||$.

Proof. Claim (i) follows from claim (ii) of Theorem 2.1.

Hereafter a pair of subscripts p, q shows the matrix size $p \times q$. The proof of claim (ii) is similar for both products AH and FA; we only cover the case of the former one.

Notice that $(AH)_{k,k} = A_{k,n}H_{n,k}$ and substitute compact SVD $A_{k,n} = S_{k,k}\Sigma_{k,k}T^T_{n,k}$ where $\Sigma_{k,k}$ is a diagonal matrix and $S_{k,k}$ and $T_{n,k}$ are orthogonal matrices. Obtain

$$(AH)_{k,k} = S_{k,k}\Sigma_{k,k}T^T_{n,k}H_{n,k} = S_{k,k}\Sigma_{k,k}G_{k,k}$$

where $G_{k,k} = T^T_{n,k}H_{n,k}$ is a $k \times k$ Gaussian matrix by virtue of Lemma 2.1. Deduce that

$$((AH)_{k,k})^+ = G^+_{k,k}\Sigma^{-1}_{k,k}S^T_{k,k}, \text{ and so } ||((AH)_{k,k})^+|| = ||G^+_{k,k}\Sigma^{-1}_{k,k}|| \le ||G^+_{k,k}||\ ||\Sigma^{-1}_{k,k}||.$$

Substitute the equations $||G^+_{k,k}|| = \nu^+_{k,k}$ and $||\Sigma^{-1}_{k,k}|| = ||A^+_{k,n}|| \le ||A^+||$ and obtain claim (ii).

Let us prove claim (iii). Notice that $(FAH)_{k,k} = F_{k,n}AH_{n,k}$, substitute compact SVD $A_{k,n} = S_{k,k}\Sigma_{k,k}T^T_{n,k}$, and obtain

$$(FAH)_{k,k} = F_{k,n}S_{k,k}\Sigma_{k,k}T^T_{n,k}H_{n,k} = G'\Sigma_{k,k}G''$$

where $G' = G_{k,n} = F_{k,n}S_{k,k}$ and $G'' = G_{n,k} = T^T_{n,k}H_{n,k}$ are Gaussian matrices by Lemma 2.1. Substitute compact SVDs $G' = S_{G',k,k}\Sigma_{G',k,k}T^T_{G',k,n}$ and $G'' = S_{G'',n,k}\Sigma_{G'',k,k}T^T_{G'',k,n}$ and obtain

$$(FAH)_{k,k} = S_{G',k,k}BT^T_{G'',k,n} \text{ for } B = \Sigma_{G',k,k}W\Sigma_{G'',k,k} \text{ and } W = T^T_{G',k,n}\Sigma_{k,k}S_{G'',n,k}.$$

Observe that the latter equation is compact SVD, and so $||W^+|| = ||\Sigma^+_{k,k}|| = ||A^+_{k,n}|| \le ||A^+||$.

Furthermore, clearly $||((FAH)_{k,k})^+|| = ||B^+||$ because $S_{G',k,k}$ and $T_{G'',k,n}$ are square orthogonal matrices. Now observe that

$$||B^+|| \le ||\Sigma^+_{G',k,k}||\ ||W^+||\ ||\Sigma^+_{G'',k,k}||$$

because $\Sigma_{G',k,k}$ and $\Sigma_{G'',k,k}$ are square diagonal matrices. Notice that

$$||\Sigma^+_{G',k,k}|| = ||G'^+|| = \nu^+_{k,n} \text{ and } ||\Sigma^+_{G'',k,k}|| = ||G''^+|| = \nu^+_{n,k}.$$

Combine the above equations for the norms $(FAH)_{k,k}$, $||W^+||$, $||\Sigma^+_{G',k,k}||$ and $||\Sigma^+_{G'',k,k}||$ and the above bound on the norm $||B^+|| = ||((FAH)_{k,k})^+||$ and obtain $||((FAH)_{k,k})^+|| \le \nu^+_{k,n}\nu^+_{n,k}||A^+_{k,n}||$. Apply this estimate to the norm $||((FAH)^T_{k,k})^+|| = ||((FAH)^T_{k,k})^+||$ and complete the proof of claim (iii) by deducing that $||((FAH)_{k,k})^+|| \le \nu^+_{k,n}\nu^+_{n,k}||A^+_{n,k}||$. □

Remark 4.2. *[PQY15, Corollary 5.2] provides a correct, although very long proof of claim (ii) of Theorem 4.3 in the case of post-multiplication by H, but states the result with an error, by writing $\nu_{n,k}$ instead of the correct $\nu_{k,k}$. We fix the statement and include a short proof for the sake of completeness of our presentation. Claim (iii) of the theorem is new and implies some important benefits of using two-sided rather than one-sided Gaussian pre-processing (see the next subsection).*

Theorems 2.2, 2.3, and 4.1–4.3 together imply the following result.

Corollary 4.1. *GENP is safe with probability 1 and is likely to be numerically safe if it is applied to the matrices FA, AH and FAH where A is a nonsingular and well-conditioned $n \times n$ matrix and F and H are Gaussian $n \times n$ matrices.*

Next we observe some additional benefits of GENP and BGE.

Remark 4.3. *Comparison of the estimates of claims (ii) and (iii) of Theorem 4.3 shows that shifting from one-sided to two-sided Gaussian pre-processing of GENP is likely to enhance its power. Indeed suppose that the integer $n - k$ is not small, say, exceeds 3. Then Theorem 2.3 implies that the norm $\nu^+_{k,k}$ is likely to exceed substantially the product $\nu^+_{k,n}\nu^+_{n,k}$, that is, GENP is substantially safer numerically with two-sided Gaussian pre-processing until the size of the GENP input has been reduced to 4×4. At this point, however, it is inexpensive to complete Gaussian elimination by means of a reliable method such as GEPP, GECP, or orthogonalization.*

Remark 4.4. *BGE can use additional benefits of block matrix algorithms and, rather surprisingly, of saving random variables and flops. E.g., first pre-process the $k \times k$ leading block of the input matrix for a proper integer $k < n$ by using $n \times k$ Gaussian multipliers. Having factored this block, decrease the input size from n to $n - k$ and then re-apply Gaussian pre-processing. Already by using such a two-step block pre-processing for $k = n/2$, we save 1/4 of all random variables and 3/8 of arithmetic operations involved.*

4.3 GENP with any nonsingular and well-conditioned pre-processing is safe and numerically safe on the average input

In the previous subsection we assumed that an input matrix A is fixed, and the multipliers F and H are Gaussian. Next we prove *a dual theorem* where we assume that the multipliers are fixed and the input matrix is Gaussian. We obtain probabilistic estimates for the matrices $(AH)_{k,k}$, $(FA)_{k,k}$, and $(FAH)_{k,k}$ similar to those of Theorem 4.3, and we will readily extend them to the estimates for pre-processed GENP applied to the average input matrix (see Corollary 4.2).

Theorem 4.4. *Assume that we are given a Gaussian $n \times n$ matrix A and a pair of $n \times n$ nonsingular and well-conditioned matrices F and H and let $\nu^+_{k,k}$ denote a random variable of Section 2.3. Then*

(i) the matrices FA, AH, and FAH are strongly nonsingular with a probability 1,

(ii) $||((AH)_{k,k})^+|| \le \nu^+_{k,k}||H^+_{n,k}||$, $||((FA)_{k,k})^+||\} \le \nu^+_{k,k}||F^+_{k,n}||$, and

(iii) $||((FAH)_{k,k})^+|| \le ||(F^+_{k,n}||\nu^+_{k,k}||H^+_{n,k}|| \le ||F^+||\nu^+_{k,k}||H^+||$.

Proof. The proof is similar to that of Theorem 4.3, and we only specify the proof of claim (iii). Recall that $(FAH)_{k,k} = F_{k,n}AH_{n,k}$, substitute compact SVDs $F_{k,n} = S_{F,k,k}\Sigma_{F,k,k}T^T_{F,n,k}$ and $H_{n,k} = S_{H,n,k}\Sigma_{H,k,k}T^T_{H,k,k}$, and obtain $(FAH)_{k,k} = S_{F,k,k}\Sigma_{F,k,k}G_{k,k}\Sigma_{H,k,k}T^T_{H,k,k}$. Here $G_{k,k} = T^T_{F,k,n}AS_{H,n,k}$ is a $k \times k$ Gaussian matrix by virtue of of Lemma 2.1. The matrices $S_{F,k,k}$, $\Sigma_{F,k,k}$, $\Sigma_{H,k,k}$, $T_{H,k,k}$ are nonsingular by assumption, and so is the matrix $G_{k,k}$ with probability 1 by virtue of claim (ii) of Theorem 2.1. Hence with probability 1, $((FAH)_{k,k})^+ = T_{H,k,k}\Sigma^{-1}_{H,k,k}G^{-1}_{k,k}\Sigma^{-1}_{F,k,k}S^T_{F,k,k}$, and so $||((FAH)_{k,k})^+|| \le ||T_{H,k,k}||\ ||\Sigma^{-1}_{H,k,k}||\ ||G^{-1}_{k,k}||\ ||\Sigma^{-1}_{F,k,k}||\ ||S^T_{F,k,k}||$. Substitute

$||T_{H,k,k}|| = ||S^T_{F,k,k}|| = 1$, $||\Sigma^{-1}_{F,k,k}|| = ||(F^+_{k,n}||$, $||\Sigma^{-1}_{H,k,k}|| = ||H^+_{n,k}||$, and $||G^{-1}_{k,k}|| = \nu^+_{k,k}$. □

Theorems 4.2, 4.3, 2.2, and 2.3 together imply the following result.

Corollary 4.2. *GENP is safe and numerically safe when it is applied to the matrices FA, AH and FAH where A is the average $n \times n$ matrix defined under the Gaussian probability distribution and F and H are $n \times n$ nonsingular and well-conditioned matrices.*

4.4 Heuristic amelioration of pre-processing for GENP

Suppose that A, F, and H are a nonsingular and well-conditioned matrix A has been pre-processed with multipliers F_i and/or H_i recursively sampled from two sufficiently large sets $\mathcal{F} = \{F_i\}_i$ and $\mathcal{H} = \{H_i\}_i$ of $n \times n$ matrices generated independently of each other and of the matrix A. Then Corollary 4.2 implies that application of GENP with pre-processing by a nonsingular and well-conditioned multipliers F_i and/or H_i is safe and numerically safe for most of nonsingular and well-conditioned input matrices A. This somewhat informal claim is in good accordance with our empirical study, although for any multipliers F we can readily exhibit bad nonsingular and well-conditioned inputs A for which GENP applied to the matrix FA fails numerically, and similarly for GENP applied to matrices AH and FAH.

Next we comment on two heuristic recipes for choosing multipliers.

(i) Clearly the product of two sparse matrices has good chances to have singular square leading blocks, and so one can be cautious about pre-processing a sparse matrix with sparse multipliers. For a partial remedy, we can more evenly distribute nonzero entries throughout a sparse multiplier, but for a more reliable remedy, we can apply dense structured multipliers.

(ii) Here is a general useful heuristic recipe for simplifying repeated pre-processing when GENP has failed numerically for two matrices AH_1 and AH_2: apply GENP to the sum $AH_1 + AH_2$ or product AH_1H_2. In our tests in Section 9, this recipe has consistently worked, but there are also other attractive options, e.g., using linear combinations or polynomials in H_1 and H_2 as multipliers.

5 Two Classes of Basic Structured Matrices for Generation of Efficient Multipliers

5.1 Matrices of discrete Fourier transform

Definition 5.1. *Write $\omega = \exp(\frac{2\pi\sqrt{-1}}{n})$, $\Omega = \Omega_n = (\omega^{ij})_{i,j=0}^{n-1}$, $\frac{1}{\sqrt{n}}\Omega$ is unitary, $\Omega^{-1} = \frac{1}{n}\Omega^H$, ω denotes a primitive n-th root of unity, Ω and Ω^{-1} denote the matrices of the discrete Fourier transform at n points and its inverse, to which we refer as DFT(n) and IDFT(n), respectively.*

Remark 5.1. *If $n = 2^k$ is a power of 2, we can apply the FFT algorithm and perform DFT(n) and IDFT(n) by using only $1.5n\log_2(n)$ and $1.5n\log_2(n) + n$ arithmetic operations, respectively. For an $n \times n$ input and any n, we can perform DFT(n) and IDFT(n) by using $cn\log(n)$ arithmetic operations, but for a larger constant c (see [P01, Section 2.3]).*

5.2 Circulant and f-circulant matrices

For a positive integer n and a complex scalar f, define the $n \times n$ unit f-circulant matrix $Z_f = \begin{pmatrix} 0 & f \\ I_{n-1} & 0 \end{pmatrix}$ and the $n \times n$ general *f-circulant matrix* $Z_f(\mathbf{v}) = \sum_{i=0}^{n} v_i Z_f^i$,

$$Z_f = \begin{pmatrix} 0 & \dots & \dots & 0 & f \\ 1 & \ddots & & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ \vdots & & \ddots & 0 & 0 \\ 0 & \dots & \dots & 1 & 0 \end{pmatrix} \quad \text{and} \quad Z_f(\mathbf{v}) = \begin{pmatrix} v_0 & fv_{n-1} & \cdots & fv_1 \\ v_1 & v_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & fv_{n-1} \\ v_{n-1} & \cdots & v_1 & v_0 \end{pmatrix}.$$

$Z = Z_0$ is the unit *down-shift matrix*. $Z_0(\mathbf{v})$ is a lower triangular Toeplitz matrix, $Z_1(\mathbf{v})$ is a *circulant* matrix. $Z_f^n = fI_n$, and the matrix $Z_f(\mathbf{v})$ is defined by its first column $\mathbf{v} = (v_i)_{i=0}^{n-1}$.

We call an f-circulant matrix a *Gaussian f-circulant* (or just *Gaussian circulant* if $f = 1$) if its first column is filled with independent Gaussian variables. For every fixed f, the f-circulant matrices form an algebra in the linear space of $n \times n$ Toeplitz matrices

$$T = (t_{i-j})_{i,j=0}^{n-1}. \tag{5.1}$$

Hereafter, for a vector $\mathbf{u} = (u_i)_{i,j=0}^{n-1}$, write $D(\mathbf{u}) = \mathrm{diag}(u_0, \dots, u_{n-1})$, that is, $D(\mathbf{u})$ is the diagonal matrix with the diagonal entries $u_0, \dots, u_{n-1}$.

Theorem 5.1. *(Cf. [P01, Theorem 2.6.4].) If $f \neq 0$, then f^n-circulant matrix $Z_{f^n}(\mathbf{v})$ of size $n \times n$ can be factored as follows, $Z_{f^n}(\mathbf{v}) = U_f^{-1} D(U_f \mathbf{v}) U_f$ for $U_f = \Omega D(\mathbf{f})$, $\mathbf{f} = (f^i)_{i=0}^{n-1}$, and $f \neq 0$. In particular, for circulant matrices, $D(\mathbf{f}) = I$, $U_f = \Omega$, and $Z_1(\mathbf{v}) = \Omega^{-1} D(\Omega \mathbf{v}) \Omega$.*

Remark 5.2. *We cannot extend this theorem directly to a $k \times k$ 0-circulant (that is, triangular Toeplitz) matrix T, but we can embed this matrix (as well as any $k \times k$ Toeplitz matrix T) into an $n \times n$ circulant matrix C for $n \geq 2k-1$ and then can readily extract the product $T\mathbf{v}$ (for any vector $\mathbf{v}$) from the product $C\mathbf{w}$ for a proper vector $\mathbf{w}$ having $\mathbf{v}$ as a subvector.*

Remark 5.3. *For an $n \times n$ Toeplitz or Toeplitz-like matrix A and an $n \times n$ circulant matrix H, one can compute a standard displacement representation of the product AH by applying just $O(n \log(n))$ flops (see definitions and derivation in [P01]).*

Corollary 5.1. *(i) Suppose that we are given a diagonal matrix $D(\mathbf{u}) = \mathrm{diag}(u_0, \dots, u_{n-1})$ for $\mathbf{u} = \Omega \mathbf{v}$. Then we can recover the vector $\mathbf{v} = \frac{1}{n} \Omega^H \mathbf{u}$, defining the circulant matrix $Z_1(\mathbf{v})$.*

(ii) If the vector $\mathbf{v}$ is Gaussian, then so is also the vector $\mathbf{u} = (u_i)_{i=1}^n = \frac{1}{\sqrt{n}} \Omega \mathbf{v}$ (by virtue of Lemma 2.1) and vice versa. Each of the two vectors defines a Gaussian circulant matrix $Z_1(\mathbf{v})$.

(iii) By choosing $u_i = \exp(\frac{\phi_i}{2\pi}\sqrt{-1})$ and real Gaussian variable ϕ_i for all i, we arrive at a random real orthogonal or unitary $n \times n$ circulant matrix $Z_1(\mathbf{v})$ defined by n real Gaussian parameters ϕ_i, $i = 0, \dots, n-1$. Alternatively we can set $\phi_i = \pm 1$ for all i and choose the signs $\pm$ at random, with i.i.d. probability 1/2 for all signs.

(iv) By adding another Gaussian parameter ϕ, we can define a random real orthogonal or unitary f-circulant matrix $Z_f(\mathbf{v})$ for $f = \exp(\frac{\phi}{2\pi}\sqrt{-1})$.

The following results imply that a Gaussian circulant matrix is likely to be well-conditioned.

Theorem 5.2. *(Cf. [PSZ15].) Suppose that $Z_1(\mathbf{v}) = \Omega^H D \Omega$ is a nonsingular circulant $n \times n$ matrix, and let $D(\mathbf{g}) = \mathrm{diag}(g_i)_{i=1}^n$, for $\mathbf{g} = (g_i)_{i=1}^n$. Then $||Z_1(\mathbf{v})|| = \max_{i=1}^n |g_i|$, $||Z_1(\mathbf{v})^{-1}|| = \min_{j=1}^n |g_j|$, and $\kappa(Z_1(\mathbf{v})) = \max_{i,j=1}^n |g_i/g_j|$, for $\mathbf{v} = \Omega^{-1}\mathbf{g}$.*

Remark 5.4. *Suppose that a circulant matrix $Z_1(\mathbf{v})$ has been defined by its first column vector $\mathbf{v}$ filled with integers ± 1 for a random choice of the i.i.d. signs $\pm$, each $+$ and $-$ chosen with probability $1/2$. Then, clearly, the entries g_i of the vector $\mathbf{g} = \Omega \mathbf{v} = (g_i)_{i=1}^n$ satisfy $|g_i| \leq n$ for all i an n, and furthermore, with a probability close to 1, $\max_{i=1}^n \log(1/|g_i|) = O(\log(n))$ as $n \to \infty$.*

Remark 5.5. *In the case of a Gaussian circulant matrix $Z_1(\mathbf{v})$, all the entries g_i are i.i.d. Gaussian variables, and the condition number $\kappa(Z_1(\mathbf{v})) = \max_{i,j=1}^n |g_i/g_j|$ is not likely to be large.*

Remark 5.6. *The superfast but numerically unstable MBA algorithm (cf. [B85], [P01, Chapter 5]) is precisely the recursive BGE, accelerated by means of exploiting the displacement structure of the input matrix throughout the recursive process of BGE. The algorithm is Fortunately, pre-processing with appropriate random structured multipliers is likely to fix these problems keeping the algorithm superfast, that is, using almost linear number of flops (cf. [P01, Sections 5.6 and 5.7]).*

6 Generation of Efficient Multipliers

6.1 What kind of multipliers do we seek?

Trying to support safe and numerically safe GENP we seek multipliers with the following properties:

1. Multipliers F and H must be nonsingular and well-conditioned.
2. The cost of the computation of the product FA, AH, FAH or FAF^H should be small.
3. Random multipliers should be generated by using fewer random variables.

4. For structured input matrices, the multipliers should have consistent structure.

5. The multipliers should enable GENP to produce accurate output with a probability close to 1 even with using no iterative refinement, at least for the inputs of reasonably small sizes.

These rules give general guidance but are a subject to trade-off: e.g., by filling multipliers with values 0, 1, and -1, we save flops when we compute their products with the input matrix A, but we increase the chances for success of our pre-processing if instead we fill the multipliers with real or complex random variables.

Some families of our new nonsingular multipliers for GENP and BGE refine (towards the above properties) the customary families of sparse and structured rectangular multipliers used for low-rank approximation (cf. [HMT11], [M11], [W14]). E.g., we sparsify the SRHT and SRFT multipliers, which substantially simplifies their generation and application as multipliers.

Remark 6.1. *Property 5 was satisfied in our tests for most of our multipliers (unlike the multipliers of [BDHT13] and [DDF14]). This property is important for $n \times n$ input matrices of smaller sizes: refinement iteration involves $O(n^2)$ flops versus cubic cost of $\frac{2}{3}n^3$ flops, involved in Gaussian elimination, but for small n quadratic cost of refinement can make up a large share of the overall cost. A single refinement iteration was always sufficient (and more frequently was not even needed) in our tests in order to match or to exceed the output accuracy of GEPP (see also similar empirical data in [PQZ13], [BDHT13], [DDF14], and [PQY15] and see [H02, Chapter 12], [GL13, Section 3.5.3], and the references therein for detailed coverage of iterative refinement).*

6.2 Some basic matrices for the generation of multipliers

We are going to use the following matrix classes in the next subsections as our building blocks for defining efficient multipliers.

1. P denotes an $n \times n$ *permutation matrix*.

2. D denotes a unitary or real orthogonal *diagonal matrix* $\mathrm{diag}(d_i)_{i=0}^{n-1}$, with fixed or random diagonal entries d_i such that $|d_i| = 1$ for all i, and so each of n entries d_i lies on the unit circle $\{x :\ |z| = 1\}$, being either nonreal or ± 1.

3. Define a *Givens rotation matrix* $G(i, j, \theta)$ (cf. [GL13]): for two integers i and j, $1 \le i < j \le n$, and a fixed or random real θ, $0 \le \theta \le 2\pi$, replace the submatrix I_2 in the ith and jth rows and columns of the identity matrix I_n by the matrix $\left(\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right)$ where $c = \cos\theta$, $s = \sin\theta$, $c^2 + s^2 = 1$.

4. Define a *Householder reflection matrix* $I_n - \frac{2\mathbf{h}\mathbf{h}^T}{\mathbf{h}^T\mathbf{h}}$ by a fixed or random vector $\mathbf{h}$ (cf. [GL13]).

We also use a DFT matrix Ω_n and transforms DFT, DFT(n) and IDFT(n) of Section 5.1 as well as circulant and f-circulant matrices of Section 5.2. Besides unitary and orthogonal diagonal matrices D, defined above, we use a nonsingular and well-conditioned diagonal matrix $\mathrm{diag}(\pm 2^{b_i})$ in Family 3 of Section 9, for random integers b_i uniformly chosen from 0 to 3.

6.3 Four families of dense structured multipliers

FAMILY 1. *The inverses of bidiagonal matrices:*

$$H = (I_n + DZ)^{-1} \text{ or } H = (I_n + Z^T D)^{-1}$$

for a diagonal matrix D and the down-shift matrix Z of Section 5.2.

We can randomize a matrix H by choosing $n-1$ random diagonal entries of the matrix D (whose leading entry makes no impact on H) and can pre-multiply it by a vector by using $2n-1$ flops.

$||H|| \le \sqrt{n}$ because nonzero entries of the triangular matrix $H = (I_n + DZ)^{-1}$ have absolute values 1, and clearly $||H^{-1}|| = ||I_n + DZ|| \le \sqrt{2}$. Hence $\kappa(H) = ||H||\ ||H^{-1}||$ (the spectral condition number of H) cannot exceed $\sqrt{2n}$ for $H = (I_n + DZ)^{-1}$, and likewise for $H = (I_n + Z^T D)^{-1}$.

FAMILY 2. *Chains of scaled and permuted Givens rotations with a DFT factor.* A permutation matrix P and a sequence of angles $\theta_1, \dots, \theta_{n-1}$ together define a permuted chain of Givens rotations

$$G(\theta_1, \dots, \theta_{n-1}) = P \prod_{i=1}^{n-1} G(i, i+1, \theta_i) \text{ for } n = 2^k.$$

Combine two such chains G_1 and G_2 with scaling and DFT into the following dense unitary matrix,

$$H = \frac{1}{\sqrt{n}} D_1 G_1 D_2 G_2 D_3 \Omega_n$$

(cf. [HMT11, Section 4.6]), for diagonal matrices D_1, D_2 and D_3. This matrix can be multiplied by a vector by using about $10n$ flops plus the cost of performing DFT. By randomizing diagonal and Givens rotation factors of H we can involve up to $7n - 2$ random variables.

FAMILY 3. *Pairs of scaled and permuted Householder reflections with a DFT factor.* Define the unitary matrix

$$H = \frac{1}{\sqrt{n}} D_1 R_1 D_2 R_2 D_3 \Omega_n$$

where D_1, D_2, and D_3 are diagonal matrices, R_1 and R_2 are Householder reflections, and Ω_n is a DFT matrix of Section 5.1. This matrix can be multiplied by a vector by using about $7n$ flops plus the cost of performing DFT. By randomizing diagonal and Householder reflection factors of H we can involve up to $6n$ random variables.

FAMILY 4. *f-circulant matrices* of Section 5.2. $H = Z_f(\mathbf{v}) = \sum_{i=0}^{n-1} v_i Z_f^i$, for the matrix Z_f of f-circular shift, defined by a scalar $f \neq 0$ and its first column vector $\mathbf{v} = (v_i)_{i=0}^{n-1}$. By randomizing this vector we can involve up to n random variables, and then Theorem 5.2 and [PSZ15] enable us to bound the condition number $\kappa(H)$. By virtue of Theorem 5.1, $Z_f(\mathbf{v}) = (FD_f)^{-1} D F D_f$ where D is a diagonal matrix, $D_f = \mathrm{diag}(f^i)_{i=0}^{n-1}$, and Ω_n is the DFT matrix of Section 5.1. Based on this expression, we can multiply the matrix H by applying two DFT(n), an IDFT(n), and additionally $n + 2\delta_f n$ multiplications and divisions where $\delta_f = 0$ if $f = 1$ and $\delta_f = 1$ otherwise.

6.4 Sparse ARSPH matrices based on Hadamard's processes

FAMILY 5. A $2^k \times 2^k$ *Abridged Recursive Scaled and Permuted Hadamard (ARSPH) matrix* $H = H_{2^k,d}$ of *depth* d has $q = 2^d$ nonzero entries in every row and in every column, for a fixed integer d, $1 \le d \le k$. Hence such a matrix is sparse unless $k - d$ is a small integer.

A special recursive structure of such a matrix allows highly efficient parallel implementation of its pre-multiplication by a vector (cf. Remark 6.2).

We recursively define matrices $H_{2^{h+1},d}$ for $h = k - d, \dots, k - 1$, as follows:

$$H_{2^{h+1},d} = D_{2^{h+1}} P_{2^{h+1}} \begin{pmatrix} H_{2^h,d} & H_{2^h,d} \\ H_{2^h,d} & -H_{2^h,d} \end{pmatrix} \bar{P}_{2^{h+1}} \bar{D}_{2^{h+1}}, \; H_{2^{k-d},d} = \begin{pmatrix} I_{2^{k-d}} & I_{2^{k-d}} \\ I_{2^{k-d}} & -I_{2^{k-d}} \end{pmatrix}. \tag{6.1}$$

Here $P_{2^{h+1}}$ and $\bar{P}_{2^{h+1}}$ are $2^{h+1} \times 2^{h+1}$ permutation matrices, $D_{2^{h+1}}$ and $\bar{D}_{2^{h+1}}$ are $2^{h+1} \times 2^{h+1}$ diagonal matrices. We can pre-multiply a matrix $H_{2^k,d}$ by a vector by using at most $3dn$ flops.

If the matrices $D_{2^{h+1}}$ or $\bar{D}_{2^{h+1}}$ are real, having nonzero entries ± 1, then these flops are additions and subtractions, and matrix $H_{2^k,d}$ is orthogonal up to scaling by a constant; otherwise it is unitary.

For random permutation matrices P_i and $\bar{P}_i$ and the diagonal matrices D_i and $\bar{D}_i$, the matrix $\widehat{B} = H_{2^k,d}$ depends on up to $(1 + 1/2 + \cdots + 1/2^d)2^{k+2} = (1 - 1/2^d)2^{k+3}$ random variables.

For $d = k$, the matrix $H_{2^k,d}$ of (6.1) turns into a dense (unabridged) RSPH matrix.

By letting $D_{2^h} = \bar{D}_{2^h} = I_{2^h}$, $P_{2^h} = \bar{P}_{2^h} = I_{2^h}$, or $D_{2^h} = \bar{D}_{2^h} = P_{2^h} = \bar{P}_{2^h} = I_{2^h}$ for all h, we arrive at the three sub-families *ARPH*, *ARSH*, and *AH* of the family of ARSPH matrices. For $d = k$, an AH matrix turns into the dense (unabridged) matrix of *Walsh-Hadamard transform.*

Special sub-families of $2^k \times 2^k$ *Abridged Scaled and Permuted Fourier (ASPF)* and *Abridged Scaled and Permuted Hadamard (ASPH)* matrices use the same initialization of (6.1),

$$\Omega_{2^{k-d},d} = H_{2^{k-d},d} = \begin{pmatrix} I_{2^{k-d}} & I_{2^{k-d}} \\ I_{2^{k-d}} & -I_{2^{k-d}} \end{pmatrix},$$

and are defined by the following recursive processes, which specialize (6.1),

$$\Omega_{2^{h+1},d} = \widehat{P}_{2^{h+1}} \begin{pmatrix} \Omega_{2^h,d} & \\ & \Omega_{2^h,d} \end{pmatrix} \begin{pmatrix} I_{2^h} & \\ & \widehat{D}_{2^h} \end{pmatrix} \begin{pmatrix} I_{2^h} & I_{2^h} \\ I_{2^h} & -I_{2^h} \end{pmatrix}, \ H_{2^{h+1},d} = \begin{pmatrix} H_{2^h,d} & H_{2^h,d} \\ H_{2^h,d} & -H_{2^h,d} \end{pmatrix}, \quad (6.2)$$

for $h = k-d, \dots, k-1$ (cf. [P01, Section 2.3] and [M11, Section 3.1]), and output the matrices $P\Omega_{2^k,d}D$ or $PH_{2^k,d}D$ for fixed or random matrices P and D of primitive types 1 and 2, respectively. Here $\widehat{D}_{2^h} = \mathrm{diag}(\omega_{2^h}^i)_{i=0}^{2^h-1}$ and $\widehat{P}_{2^h}$ is the $2^h \times 2^h$ odd/even permutation matrix, such that $\widehat{P}_{2^h}(\mathbf{u}) = \mathbf{v}$, $\mathbf{u} = (u_i)_{i=0}^{2^h-1}$, $\mathbf{v} = (v_i)_{i=0}^{2^h-1}$, $v_j = u_{2j}$, $v_{j+2^{h-1}} = u_{2j+1}$, and $j = 0, 1, \dots, 2^{h-1}-1$.

The sub-families of ASPF and ASPH matrices in turn have sub-families of *ASF, APF, ASH, and APH* matrices. The $n \times n$ AF matrix for $d = k$ is just the matrix $DFT(n)$, Ω_n of Definition 5.1.

Recursive process (6.2) defining the matrix Ω_n is known as the decimation in frequency (DIF) radix-2 representation of FFT; transposition turns it into the decimation in time (DIT) radix-2 representation of FFT. The numbers of random variables involved into generation of general ARSPF and ARSPH matrices decrease to at most $(1 - 1/2^d)2^{k+1}$ for ASPF and ASPH matrices, further decrease to at most $(1 - 1/2^d)2^{k-1}$ for ASF, APF, ASH, and APH matrices, and the AF and AH matrices involve no random variables. The estimated arithmetic cost of pre-multiplication of all these submatrices by a vector is the same as in the case of ARSPF and ARSPH matrices.

Another well-known special case is given by recursive two-sided Partial Random Butterfly Transforms (PRBTs), based on two unpublished Technical Reports of 1995 by Parker and by Parker and Pierce, but improved, carefully implemented, and then extensively tested in [BDHT13].

For an $n \times n$ input matrix and even $n = 2q$, that paper defines PRBT as follows,

$$B^{(n)} = \frac{1}{\sqrt{2}} \begin{pmatrix} R & S \\ R & -S \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} I_q & I_q \\ I_q & -I_q \end{pmatrix} \mathrm{diag}(R, S) \quad (6.3)$$

where R and S are random diagonal nonsingular matrices. The paper [BDHT13] defines multipliers F and H recursively by using PRBT blocks. According to [BDHT13], the two-sided recursive processes of depth $d = 2$ with PRBT blocks are "sufficient in most cases". In such processes $F = \mathrm{diag}(B_1^{(n/2)}, B_2^{(n/2)})B^{(n)}$, and the multiplier H is defined similarly. In the case of depth-d recursion, $d \geq 2$, each of the multipliers F and H is defined as the product of d factors made up of 2^j diagonal blocks of size $n/2^j \times n/2^j$, for $j = 0, \dots, d-1$, each block of the same type as above, and the two-sided multiplication by a vector involves $6dn$ flops for $n = 2^k$.

6.5 Four additional families of sparse multipliers

FAMILY 6. *Sparse f-circulant matrices $H = Z_f(\mathbf{v})$* are defined by a fixed or random scalar f, $|f| = 1$, and the first column $\mathbf{v}$ having exactly q nonzero entries, for $q \ll n$. The positions and values of non-zeros can be randomized (and then the matrix depends on up to $2n+1$ random variables).

Such a matrix can be pre-multiplied by a vector by using at most $(2q-1)n$ flops or, in the real case where $f = \pm 1$ and $v_i = \pm 1$ for all i, by using at most qn additions and subtractions.

FAMILY 7. Abridged *f^n-circulant $n \times n$ matrices H* of a recursion depth d for a scalar f, $n = 2^k$, and two integers d and k such that $|f| = 1$, $k > d \geq 0$, and the integer $k - d$ is not small,

$$H = (MD_f)^{-1} DMD_f.$$

Here $D = \mathrm{diag}(d_i)_{i=1}^n$ and $D_f = \mathrm{diag}(f^i)_{i=0}^{n-1}$ are diagonal matrices, $|d_i| = 1$ for all i and M is an AF or AH matrix of Family 5 of recursion depth d.

Such a matrix H is unitary up to scaling by a constant (or orthogonal if the matrices M, D and D_f are real) and can be pre-multiplied by a vector by using at most $6dn$ flops. It involves up to n random variables, but would involve $3n$ (resp. 2n) such variables if we use an ASPF or ASPH (resp. ASF, ASH, ASF, or APH) rather than an AF or AH matrix F. For $d = k$ the AF matrix M turns into the DFT matrix Ω_n and H turns into the g-circulant matrix (cf. Theorem 5.1)

$$Z_g(\mathbf{v}) = D_f^{-1}\Omega_n^H D\Omega_n D_f, \text{ for } g = f^n,\ D_f = \mathrm{diag}(f^i)_{i=0}^{n-1}, D = \mathrm{diag}(d_i)_{i=0}^{n-1}, \text{ and } (d_i)_{i=0}^{n-1} = \Omega_n D_f \mathbf{v}.$$

For $f = 1$, the above expressions are simplified: $g = 1$, $D_f = I_n$, $M = \Omega_n$, and $H = \sum_{i=0}^{n-1} v_i Z_1^i$ is a circulant matrix $Z_1(\mathbf{v}) = \Omega_n^H D\Omega_n,\ D = \mathrm{diag}(d_i)_{i=0}^{n-1},\ \text{for } (d_i)_{i=0}^{n-1} = \Omega_n \mathbf{v}$.

FAMILY 8. *Scaled and permuted chains of Givens rotations with an AF or AH factor.* A permutation matrix P and $n-1$ angles $\theta_1, \dots, \theta_{n-1}$ together define a permuted chain of Givens rotations,

$$G(\theta_1, \dots, \theta_{n-1}) = P \prod_{i=1}^{n-1} G(i, i+1, \theta_i) \text{ for } n = 2^k.$$

One can combine two such chains G_1 and G_2 with scaling and DFT into the dense unitary matrix

$$H = D_1 G_1 D_2 G_2 D_3 \Omega_n$$

(cf. [HMT11, Section 4.6]), for diagonal matrices D_1, D_2 and D_3 of primitive type 2.

For $n = 2^k \gg 2^d$ we can make matrix H sparse by replacing the factor Ω_n with an $n \times n$ AF or AH matrix; then we can pre-multiply this matrix by a vector by using $(1.5d + 10)n$ flops.

A matrix H involves up to $7n$ random variables, but we can increase this bound by $2n$ (resp. by n) if we replace the factor Ω_n with an ASPF or ASPH (resp. by APF, APH, ASF, or ASH) rather than AF or AH matrix.

FAMILY 9. *Pairs of scaled and permuted Householder reflections with an AF or AH factor.* Define the unitary matrix

$$H = \frac{1}{\sqrt{n}} D_1 R_1 D_2 R_2 D_3 \Omega_n$$

where D_1, D_2, and D_3 are diagonal matrices, R_1 and R_2 are Householder reflections, and Ω_n is a DFT matrix. This matrix can be multiplied by a vector by using about $7n$ flops plus the cost of performing DFT. By randomizing diagonal, permutation and Householder reflection matrices we can involve up to $7n$ random variables. For $n = 2^k$ and sparse vectors $\mathbf{h}_i$ defining Householder reflections R_i, for $i = 1, 2$, we can make matrix H sparse by replacing the factor Ω_n with an $n \times n$ AF or AH matrix of recursion depth d such that $2^d \ll n$; then we can multiply this matrix by a vector by using at most $(1.5d + 7)n$ flops.

6.6 Estimated numbers of random variables and flops

Table 6.1 summarizes upper bounds on (a) the numbers of random variables involved into the matrices $\widehat{B}$ of Families 1–9 and (b) the numbers of flops for multiplication of such a matrix $\widehat{B}$ by a vector.[7] Compare these data with n^2 random variables and $(2n-1)n$ flops involved in the case of a Gaussian $n \times n$ multiplier and see more refined bounds in Sections 6.3–6.5.

Remark 6.2. *Other observations besides flop estimates can be decisive. E. g., a special recursive structure of an ARSPH matrix H_{2^k} allows highly efficient parallel implementation of its multiplication by a vector based on Application Specific Integrated Circuits (ASICs) and Field-Programmable Gate Arrays (FPGAs), incorporating Butterfly Circuits [DE].*

[7] The matrices of Families 2–4 involve up to $7n - 2$, $7n$, and n random variables, respectively, and are multiplied by a vector by using $O(n \log(n))$ flops.

Table 6.1: The numbers of random variables and flops

Family	1	5	6	7	8	9
random variables	$n-1$	2^{k+3}	$2q+1$	n	$7n-2$	$7n$
flops	$2n-1$	$3dn$	$(2q-1)n$	$6dn$	$(1.5d+10)n$	$(1.5d+7)n$

6.7 Other basic families of multipliers

There is a variety of other interesting basic matrix families. E.g., one can generalize Family 6 to the family of sparse matrices with q nonzeros entries ± 1 in every row and in every column for a fixed integer q, $1 \le q \ll n$. Such matrices can be defined as the sums $\sum_{i=1}^{q} \widehat{D}_i P_i$ for fixed or random permutation matrices P_i and diagonal matrices $\widehat{D}_i$, involve up to qn random variables, and can be pre-multiplied by a vector at the same estimated cost as sparse f-circulant matrices.

For another example, one can modify a Givens chains of the form $D_1 G_1 D_2 G_2 D_3 F$, for F denoting an DFT, AH, ASPF, ASPH, APF, APH, ASF, or ASH matrix, by replacing one of the matrices G_1 or G_2 with a permuted Householder reflection matrix or the inverse of a bidiagonal matrix.

The reader can find more families of multipliers in our Section 9.

7 Symbolic and numerical GENP with one-sided randomized circulant pre-processing

7.1 GENP with one-sided randomized circulant pre-processing is likely to be safe universally (for any input)

In symbolic application of GENP one only cares about its safety rather than numerical safety. In this case the power of randomization is strengthened. Claims (i) of Theorems 4.3 and 4.4 imply that GENP pre-processed with any nonsingular multiplier F or H (e.g., $H = I_n$) or with any pair of such multipliers is unsafe only for an algebraic variety of lower dimension in the space of all inputs and that for a fixed input GENP is safe if pre-processed with almost any nonsingular multiplier or any pair of such multipliers apart from an algebraic variety of lower dimension. Therefore in symbolic computations one can quite confidently apply GENP with no pre-processing and in the very unlikely case of failure re-apply GENP with a random nonsingular multiplier.

For a theoretical challenge, however, one can seek randomized multipliers that support safe GENP and BGE universally, that is, with probability 1 for any nonsingular input. This challenge has been met already in 1991 (see, e.g., [BP94, Section 2.13, entitled "Regularization of a Matrix via Preconditioning with Randomization"]). Among the known options, one-sided pre-processing with random Toeplitz multipliers of [KP91] is most efficient, but a little inferior two-sided pre-processing with random triangular Toeplitz multipliers of [KS91] has been more widely advertised in the Computer Algebra community and has become more popular there.

Next we prove that even pre-processing with one-sided Gaussian or random uniform circulant multipliers (see Section 5.2 for definitions) is likely to make GENP safe, that is, involving no divisions by 0. Using circulant multipliers saves 50% of random variables and enables a 4-fold (resp. 2-fold) acceleration of the pre-processing of [KS91] (resp. [KP91]).

We need more than two pages besides the space for definitions in order to prove that result, but it enables us to improve substantially the popular and decades-old recipes for pre-processing GENP for symbolic computations. Namely, pre-processing of [KS91] requires pre- and post-multiplication of an $n \times n$ input matrix A by an upper and a lower triangular Toeplitz matrices, respectively, at the overall cost dominated by the cost of performing twelve DFT(n) per row of an input matrix A (see Remark 5.2), and in addition one must generate $2n-1$ random variables. Pre-processing of [KP91] uses as many random variables and six DFT(n) per row of A. Our present algorithm only post- or pre-multiplies a matrix A by a single circulant matrix, at the cost dominated by the cost of

performing three DFT(n) per row of an input matrix A, and uses only n random variables.

Let us supply the details.

Theorem 7.1. *Suppose $A = (a_{i,j})_{i,j=1}^{n}$ is a nonsingular matrix, $T = (t_{i-j+1})_{i,j=1}^{n}$ is a Gaussian f-circulant matrix, $B = AT = (b_{i,j})_{i,j=1}^{n}$, f is a fixed complex number, $t_1, \dots, t_n$ are variables, and $t_k = f t_{n+k}$ for $k = 0, -1, \dots, 1-n$. Let $B_{l,l}$ denotes the l-th leading blocks of the matrix B for $l = 1, \dots, n$, and so $\det(B_{l,l})$ are polynomial in $t_1, \dots, t_n$, for all l, $l = 1, \dots, n$. Then neither of these polynomials vanishes identically in $t_1, \dots, t_n$.*

Proof. Fix a positive integer $l \le n$. With the convention $\alpha_{k\pm n} = f\alpha_k$, for $k = 1, \dots, n$, we can write

$$B_{l,l} = \Big(\sum_{k_1=1}^{n} \alpha_{k_1} t_{k_1}, \sum_{k_2=1}^{n} \alpha_{k_2+1} t_{k_2}, \dots, \sum_{k_l=1}^{n} \alpha_{k_l+l-1} t_{k_l} \Big), \tag{7.1}$$

where α_j is the jth column of $A_{l,n}$. Let $a_{i,j+n} = f a_{i,j}$, for $k = 1, \dots, n$, and readily verify that

$$b_{i,j} = \sum_{k=1}^{n} a_{i,j+k-1} t_k,$$

and so $\det(B_l)$ is a homogeneous polynomial in $t_1, \dots, t_n$.

Now Theorem 7.1 is implied by the following lemma.

Lemma 7.1. *If $\det(B_{l,l}) = 0$ identically in all the variables $t_1, \dots, t_n$, then*

$$\det(\alpha_{i_1}, \alpha_{i_2}, \dots, \alpha_{i_l}) = 0 \tag{7.2}$$

for all l-tuples of subscripts $(i_1, \dots, i_l)$ such that $1 \le i_1 < i_2 < \cdots < i_l \le n$.

Indeed let $A_{l,n}$ denote the block submatrix made up of the first l rows of A. Notice that if (7.2) holds for all l-tuples of the subscripts $(i_1, \dots, i_l)$ above, then the rows of the block submatrix $A_{l,n}$ are linearly dependent, but they are the rows of the matrix A, and their linearly dependence contradicts the assumption that the matrix A is nonsingular.

In the rest of this section we prove Lemma 7.1. At first we order the l-tuples $I = (i_1, \dots, i_l)$, each made up of l positive integers written in nondecreasing order, and then we apply induction.

We order all l-tuples of integers by ordering at first their largest integers, in the case of ties by ordering their second largest integers, and so on.

We can define the classes of these l-tuples up to permutation of their integers and congruence modulo n, and then represent every class by the l-tuple of nondecreasing integers between 1 and n. Then our ordering of l-tuples of ordered integers takes the following form, $(i_1, \dots, i_l) < (i'_1, \dots, i'_l)$ if and only if there exist a subscript j such that $i_j < i'_j$ and $i_k = i'_k$ for $k = j+1, \dots, l$.

We begin our proof of Lemma 7.1 with the following basic result.

Lemma 7.2. *It holds that*

$$\det(B_{l,l}) = \sum_{1 \le i_1 \le i_2 \le \cdots \le i_l \le n} a_{\prod_{j=1}^{l} t_{i_j}} \prod_{j=1}^{l} t_{i_j}$$

where a tuple $(i_1, \dots, i_l)$ may contain repeated elements,

$$a_{\prod_{j=1}^{l} t_{i_j}} = \sum_{(i'_1, \dots, i'_l)} \det(\alpha_{i'_1}, \alpha_{i'_2+1}, \dots, \alpha_{i'_l+l-1}), \tag{7.3}$$

and $(i'_1, \dots, i'_l)$ ranges over all permutations of $(i_1, \dots, i_l)$.

Proof. By using (7.1) we can expand $\det(B_{l,l})$ as follows,

$$\begin{aligned}
\det(B_{l,l}) &= \det\Big(\sum_{k_1=1}^{n}\alpha_{k_1}t_{k_1}, \sum_{k_2=1}^{n}\alpha_{k_2+1}t_{k_2}, \ldots, \sum_{k_l=1}^{n}\alpha_{k_l+l-1}t_{k_l}\Big) \\
&= \sum_{i_1=1}^{n} t_{i_1} \det\Big(\alpha_{i_1}, \sum_{k=1}^{n}\alpha_{k+1}t_k, \ldots, \sum_{k_l=1}^{n}\alpha_{k_l+l-1}t_{k_l}\Big) \\
&= \sum_{i_1=1}^{n} t_{i_1} \sum_{i_2=1}^{n} t_{i_2} \det\Big(\alpha_{i_1}, \alpha_{i_2+1}, \sum_{k_2=1}^{n}\alpha_{k_2+2}t_{k_2}, \ldots, \sum_{k_l=1}^{n}\alpha_{k_l+l-1}t_{k_l}\Big) \\
&= \ldots \\
&= \sum_{i_1=1}^{n} t_{i_1} \sum_{i_2=1}^{n} t_{i_2} \cdots \sum_{i_l=1}^{n} t_{i_l} \det\left(\alpha_{i_1}, \alpha_{i_2+1}, \ldots, \alpha_{i_l+l-1}\right). \qquad (7.4)
\end{aligned}$$

Consequently the coefficient $a_{\prod_{j=1}^{l} t_{i_j}}$ of any term $\prod_{j=1}^{l} t_{i_j}$ is the sum of all determinants

$$\det\left(\alpha_{i'_1}, \alpha_{i'_2+1}, \ldots, \alpha_{i'_l+l-1}\right)$$

where $(i'_1, \ldots, i'_l)$ ranges over all permutations of $(i_1, \ldots, i_l)$, and we arrive at (7.3).

□

In particular, the coefficient of the term t_1^l is $a_{t_1 \cdot t_1 \cdot \ldots \cdot t_1} = \det(\alpha_1, \alpha_2, \ldots, \alpha_l)$. This coefficient equals zero because $B_{l,l}$ is identically zero, by assumption of lemma 7.1, and we obtain

$$\det(\alpha_1, \alpha_2, \ldots, \alpha_l) = 0. \qquad (7.5)$$

This is the basis of our inductive proof of Lemma 7.1. In order to complete the induction step, it remains to prove the following lemma.

Lemma 7.3. *Let $J = (i_1, \ldots, i_l)$ be a tuple such that $1 \le i_1 < i_2 < \cdots < i_l \le n$.*
Then J is a subscript tuple of the coefficient of the term $\prod_{j=1}^{l} t_{i_j - j + 1}$ in equation (7.3).
Moreover, J is the single largest tuple among all subscript tuples.

Proof. Hereafter $\det(\alpha_{i'_1}, \alpha_{i'_2+1}, \ldots, \alpha_{i'_l+l-1})$ is said to be *the determinant associated with the permutation* $(i'_1, \ldots, i'_l)$ of $(i_1, \ldots, i_l)$ in (7.3). Observe that $\det(\alpha_{i_1}, \ldots, \alpha_{i_l})$ is the determinant associated with $\mathcal{I} = (i_1, i_2 - 1, \ldots, i_l - l + 1)$ in the coefficient $a_{\prod_{j=1}^{l} t_{i_j - j + 1}}$.

Let $\mathcal{I}'$ be a permutation of $\mathcal{I}$. Then $\mathcal{I}'$ can be written as $\mathcal{I}' = (i_{s_1} - s_1 + 1, i_{s_2} - s_2 + 1, \ldots, i_{s_l} - s_l + 1)$, where $(s_1, \ldots, s_l)$ is a permutation of $(1, \ldots, l)$. The determinant associated with $\mathcal{I}'$ has the subscript tuple $\mathcal{J}' = (i_{s_1} - s_1 + 1, i_{s_2} - s_2 + 2, \ldots, i_{s_l} - s_l + l)$. j satisfies the inequality $j \le i_j \le n - l + j$ because by assumption $1 \le i_1 < i_2 < \cdots < i_l \le n$, for any $j = 1, 2, \ldots, l$. Thus, $i_{s_j} - s_j + j$ satisfies the inequality $j \le i_{s_j} - s_j + j \le n - l + j \le n$, for any s_j. This fact implies that no subscript of $\mathcal{I}'$ is negative or greater than n.

Let $\mathcal{J}'' = (i_{s_{r_1}} - s_{r_1} + r_1, i_{s_{r_2}} - s_{r_2} + r_2, \ldots, i_{s_{r_l}} - s_{r_l} + r_l)$ be a permutation of $\mathcal{J}$ such that its elements are arranged in the nondecreasing order. Now suppose $\mathcal{J}'' \ge J$. Then we must have $i_{s_{r_l}} - s_{r_l} + r_l \ge i_l$. This implies that

$$i_l - i_{s_{r_l}} \le r_l - s_{r_l}. \qquad (7.6)$$

Observe that

$$l - s_{r_l} \le i_l - i_{s_{r_l}} \qquad (7.7)$$

because $i_1 < i_2 < \cdots < i_l$ by assumption. Combine bounds (7.6) and (7.7) and obtain that $l - s_{r_l} \le i_l - i_{s_{r_l}} \le r_l - s_{r_l}$ and hence $r_l = l$.

Apply this argument recursively for $l - 1, \ldots, 1$ and obtain that $r_j = j$ for any $j = 1, \ldots, l$. Therefore $\mathcal{J} = \mathcal{J}'$ and $\mathcal{I}' = \mathcal{I}$. It follows that J is indeed the single largest subscript tuple. □

By combining Lemmas 7.2 and 7.3, we support the induction step of the proof of Lemma 7.1, which we summarize as follows:

Lemma 7.4. *Assume the class of l-tuples of l positive integers written in the increasing order in each l-tuple and write* $\det(I) = \det(\alpha_{i_1}, \alpha_{i_2}, \dots, \alpha_{i_l})$ *if* $I = (\alpha_{i_1}, \alpha_{i_2}, \dots, \alpha_{i_l})$.

Then $\det(I) = 0$ *provided that* $\det(J) = 0$ *for all* $J < I$.

Finally we readily deduce Lemma 7.2 by combining this result with equation (7.5). This completes the proof of Theorem 7.1. □

Corollary 7.1. *Assume any nonsingular $n \times n$ matrix A and a finite set $\mathcal{S}$ of cardinality $|\mathcal{S}|$. Sample the values of the n coordinates $v_1, \dots, v_n$ of a vector $\mathbf{v}$ at random from this set. Fix a complex f and define the matrix $H = Z_f(\mathbf{v})$ of size $n \times n$, with the first column vector $\mathbf{v} = (v_i)_{i=1}^n$. Then GENP and BGE are safe for the matrix AH*

(i) with a probability of at least $1 - 0.5(n-1)n/|\mathcal{S}|$ if the values of the n coordinates $v_1, \dots, v_n$ of a vector $\mathbf{v}$ have been sampled uniformly at random from a finite set $\mathcal{S}$ of cardinality $|\mathcal{S}|$ or

(ii) with probability 1 if these coordinates are i.i.d. Gaussian variables.

(iii) The same claims hold for the matrix FA.

Proof. Theorems 4.1, 7.1, and 2.1 together imply claims (i) and (ii) of the corollary. By applying transposition to the matrix AH, extend the results to claim (iii). □

7.2 GENP with any one-sided circulant pre-processing fails numerically for some specific inputs

By virtue of Corollary 7.1 random circulant pre-processing is a universal means for ensuring safe GENP, but is it also a universal means for ensuring numerically safe GENP?

By virtue of [P16, Corollary 6.3.1], the answer is "No",[8] and moreover GENP is numerically unsafe when it is applied to the DFT matrix Ω_n already for a reasonably large integer n as well as to the matrices $\Omega_n Z_1(\mathbf{v})$ and any vector $\mathbf{v}$ of dimension n, that is, for any circulant matrix $Z_1(\mathbf{v})$, and consequently for a random circulant matrix $Z_1(\mathbf{v})$. By combining the proof of [P16, Corollary 6.3.1] with Theorem 5.1 one can immediately extend the result to any f-circulant multiplier $Z_f(\mathbf{v})$ for any $f \neq 0$. It follows that GENP also fails numerically for the input pairs (A, H) where $A = \Omega_n Q$ and $H = Q^H Z_f(\mathbf{v})$ for any unitary matrix Q.

Surely one does not need to use GENP in order to solve a linear system of equations with a DFT coefficient matrix, but the above results reveal the difficulty in finding universal classes of structured pre-processing for GENP. Having specific bad pairs of inputs and multipliers does not contradict claim (ii) of Corollary 4.1, and actually in extensive tests in [PQZ13] and [PQY15] very good numerical stability has been observed when we applied GENP to various classes of nonsingular well-conditioned input matrices with random circulant multipliers.

8 Alternatives: augmentation and additive pre-processing

Other randomization techniques, besides multiplications, are also beneficial for various fundamental matrix computations (see [M11], [HMT11], [PGMQ], [PIMR10], [PQ10], [PQZC], [PQ12], [PZa], and the bibliography therein). By combining our present results with those of [PZa], we next supports GENP and BGE by means of *randomized augmentation* of a matrix, that is, of appending to it random rows and columns, as well as by means of an alternative and closely related technique of *additive pre-processing* (cf. (8.2)–(8.4)).

[8] This nontrivial result has been deduced by using the recent specialization in [P15] to Cauchy and Vandermonde matrices of the general techniques of transformation of matrix structure proposed in [P90].

8.1 Gaussian augmentation

By virtue of claim (i) of [PZa, Theorem 10.1], properly scaled Gaussian augmentation of a sufficiently large size is likely to produce strongly nonsingular and strongly well-conditioned matrices. Namely, this holds when we augment an $n \times n$ matrix A and produce the matrix $K = \begin{pmatrix} I_h & V^T \\ U & A \end{pmatrix}$. Here U and V are $n \times h$ Gaussian matrices filled with $2hn$ i.i.d. Gaussian random entries and $\nu \le h \le n$, for ν denoting an upper bound on the numerical nullity of the leading blocks, that is, on their numerical co-rank. In the dual version of that theorem, average matrix K is strongly nonsingular and strongly well-conditioned if A is a Gaussian matrix and if the matrices U and V have full rank and are scaled so that $||U|| \approx ||V|| \approx 1$, are well-conditioned.

8.2 SRFT augmentation

By virtue of [PZa, Section 8] we are likely to succeed if we apply GENP to the matrix K above obtained by augmenting a nonsingular and well-conditioned matrix A with SRFT matrices U and V of a sufficiently large size replacing the Gaussian ones of the previous subsection.[9]

Let us supply some details. Claim (ii) of [PZa, Theorem 10.1] implies that we are likely to produce a strongly nonsingular and strongly well-conditioned matrix K if U and V are SRFT matrices such that $\nu \ge q = cn$ for a sufficiently large constant c and if

$$4\Big(\sqrt{\nu} + \sqrt{8\log_2(\nu n)}\Big)^2 \log_2(\nu) \le h. \tag{8.1}$$

Indeed, the $q \times q$ leading blocks $K_{q,q}$ of the matrix K are the identity matrices I_q for $q = 1, \dots, h$, and so we only need to estimate the probability that the leading blocks $K_{q,q}$ are well-conditioned for all $q > h$ because their nonsingularity with probability 1 readily follows from Theorem 2.1.

It is proven in [PZa, Section 8] that under (8.1) this property holds with a probability at least $1 - c'/q$ for a constant c' and a fixed q. Therefore it holds for all q, $q = h+1, \dots, h+n$ with a probability at least $1 - c' \sum_{q=h+1}^{h+n} 1/q \approx 1 - c' \ln(\frac{h+n}{h+1}) = 1 - c' \ln(1 + \frac{n-1}{h+1}) \ge 1 - c' \ln(1 + 1/c)$. This is close to 1 for a sufficiently large constant c, and our claim about the matrix K follows.

8.3 Linking augmentation to additive pre-processing

Consider an augmented matrix K and its inverse K^{-1}. Then its $n \times n$ trailing (that is, southeastern) block is C^{-1} for $C = A - UV^T$. Indeed

$$K = \begin{pmatrix} I_h & O_{h,n} \\ U & I_n \end{pmatrix} \begin{pmatrix} I_h & O_{h,n} \\ O_{n,h} & C \end{pmatrix} \begin{pmatrix} I_h & V^T \\ O_{n,h} & I_n \end{pmatrix}. \tag{8.2}$$

Consequently

$$K^{-1} = \begin{pmatrix} I_h & -V^T \\ O_{n,h} & I_n \end{pmatrix} \begin{pmatrix} I_h & O_{h,n} \\ O_{n,h} & C^{-1} \end{pmatrix} \begin{pmatrix} I_h & O_{h,n} \\ -U & I_n \end{pmatrix}, \tag{8.3}$$

$C^{-1} = \mathrm{diag}(O_{h,h} I_n) K^{-1} \mathrm{diag}(O_{h,h} I_n)$, and the claim follows.

Now deduce that $||K^{-1}||/N \le ||C^{-1}|| \le ||K^{-1}||$, $||K||/N \le ||C^{-1}|| \le N\,||K||$, and hence

$$\kappa(K)/N^2 \le \kappa(C) \le N\kappa(K), \text{ for } N = (1 + ||U||)(1 + ||V||). \tag{8.4}$$

Equations (8.2)-(8.4) closely link augmentation $A \to K$ with *additive pre-processing* $A \to C$ and also link the leading blocks of the augmented matrices with those output by additive pre-processing.

Indeed readily extend our observations to obtain that the $k \times k$ trailing submatrix of the leading block $K^{-1}_{h+k,h+k}$ of the matrix K is the $k \times k$ leading block $C^{-1}_{k,k}$ of the matrix C^{-1} and that

$$\kappa(K_{h+k,h+k})/N^2 \le \kappa(C_{k,k}) \le N\kappa(K_{h+k,h+k}), \tag{8.5}$$

for $k = 1, \dots, n$. If the factor N is reasonably bounded, which is likely for Gaussian matrices U and V, then the matrix $C_{k,k}$ is nonsingular and well-conditioned if and only if so is the matrix $K_{h+k,h+k}$.

[9] The same results and estimates hold if one substitutes matrices of SRHT for the ones of SRFT (cf. [T11]).

8.4 Transition back to computations with the original matrix. Expansion and homotopy continuation

Having computed the inverses K^{-1} and C^{-1} by applying GENP or BGE to the augmented matrix K, one can simplify computation of the inverse A^{-1} of the original matrix A by applying the Sherman–Morrison–Woodbury formula[10]

$$A^{-1} = C^{-1} - C^{-1}U(I_h + V^TC^{-1}U)^{-1}V^TC^{-1}, \tag{8.6}$$

for $C = A - UV^T$ (cf. [GL13, page 65]).

Computing the inverse A^{-1} by means of SMW formula (8.6) may still cause numerical problems at the stages of computing and inverting the matrix $I_h + V^TC^{-1}U$, but they are less likely to occur if the matrix C is nonsingular and well-conditioned, which we expect to be the case in this application.

In order to strengthen the chances for the success of this approach we can apply some heuristic recipes. In our tests with benchmark inputs, we succeeded by simply doubling the lower bound ν on the dimension h of additive pre-processing or equivalently by keeping the same bound ν but requiring that $2\nu \le h \le n$. Another natural remedy is the well-known general technique of *homotopy continuation*, with which one proceeds as follows. Fix two matrices U and V as before and define the matrices $C(\tau) = A - \tau UV^T$ and $S(\tau) = I_h + \tau V^TC(\tau)^{-1}U$ for a nonnegative parameter τ. Suppose that the matrix $S(\bar\tau)$ is diagonally dominant for some positive $\bar\tau$. Notice that the matrices $C(1) = A - UV^T$ and $S(0) = I_h$ are readily invertible, fix the decreasing sequence of the values τ_k, $k = 0, 1, \dots, l$, such that $\tau_0 = 1 > \tau_1 > \cdots > \tau_l = 0$, and compute the sequence of the matrices $\tau_j V^TC(\tau_j)^{-1}U$, for $j = 0, 1, \dots, l$, by extending SMW formula (8.6) as follows,

$$C(\tau_{j+1})^{-1} = C(\tau_j)^{-1} - \Delta_j C(\tau_j)^{-1}U(I_h + \Delta_j V^TC(\tau_j)^{-1}U)^{-1}V^TC(\tau_j)^{-1},$$

for $\Delta_j = \tau_{j+1} - \tau_j$. For sufficiently small values Δ_j, the matrices $I_h + \Delta_j V^TC(\tau_j)^{-1}U$ are diagonally dominant and readily invertible, and then we can numerically safely perform l homotopy continuation steps for the transition from the inverse $C(1)^{-1} = C^{-1}$ to $C(0)^{-1} = A^{-1}$.

By generalizing SMW formula (8.6) and writing $C = A - UV^T$, we can readily express the inverse K^{-1} of the augmented matrix K,

$$K^{-1} = \begin{pmatrix} I_h & -V^TC^{-1} \\ O_{n,h} & C^{-1} \end{pmatrix} \begin{pmatrix} I_h & O_{h,n} \\ -U & I_n \end{pmatrix}, \ K = \begin{pmatrix} I_h & V^T \\ U & A \end{pmatrix} = \begin{pmatrix} I_h & O_{h,n} \\ U & I_n \end{pmatrix} \begin{pmatrix} I_h & V^T \\ O_{n,h} & C \end{pmatrix}.$$

9 Numerical Experiments

Numerical experiments have been performed, by using MATLAB, by the second author in the Graduate Center of the City University of New York on a Dell computer with the Intel Core 2 2.50 GHz processor and 4G memory running Windows 7. Gaussian matrices have been generated by applying the standard normal distribution function randn of MATLAB. We refer the reader to [PQZ13], [PQY15], and [PZa] for other extensive tests of GENP with randomized pre-processing.

Tables 9.1–9.4 show the maximum, minimum and average relative residual norms $||A\mathbf{y} - \mathbf{b}||/||\mathbf{b}||$ as well as the standard deviation for the solution of 1000 linear system $A\mathbf{x} = \mathbf{b}$ with Gaussian vector $\mathbf{b}$ and $n \times n$ input matrix A for each n, $n = 256, 512, 1024$ and 10 linear systems for $n = 2048, 4096$,

$$A = \begin{pmatrix} A_k & B \\ C & D \end{pmatrix}, \tag{9.1}$$

with $k \times k$ blocks A_k, B, C and D, for $k = n/2$, scaled so that $||B|| \approx ||C|| \approx ||D|| \approx 1$, the $k - 4$ singular values of the matrix A_k were equal 1 and the other ones were set to 0 (cf. [H02, Section 28.3]), and with Gaussian Toeplitz matrices B, C, and D, that is, with Toeplitz matrices of (5.1), each defined by the i.i.d. Gaussian entries of its first row and first column. (The norm $||A^{-1}||$ ranged from 2.2×10^1 to 3.8×10^6 in these tests.) The linear systems have been solved by using GEPP,

[10] Hereafter we use the acronym *SMW*.

GENP, or GENP pre-processed with real Gaussian, real Gaussian circulant, and random circulant multipliers, each followed by a single loop of iterative refinement.

In the tests covered in Table 9.4, the matrix A was set to equal Ω, the matrix of DFT(n). For pre-processing, either Gaussian or Gaussian unitary circulant matrices $C = \Omega^{-1}D(\Omega\mathbf{v})\Omega$ have been used as multipliers, with $\mathbf{v} = (v_i)_{i=0}^{n-1}$, $v_i = \exp(2\pi\phi_i\sqrt{-1}/n)$ and n i.i.d. real Gaussian variables ϕ_i, $i = 0, \dots, n-1$ (cf. Theorem 5.1 and Remark 5.5).

As should be expected, GEPP has always produced accurate solutions, with the average relative residual norms ranging from 10^{-12} to 7×10^{-13}, but GENP with no pre-processing has consistently produced corrupted output with relative residual norms ranging from 10^{-3} to 10^{2} for the input matrices A of equation (9.1). Even much worse was the output accuracy when GENP with no pre-processing or with Gaussian circulant pre-processing was applied to the matrix $A = \Omega$. In all other cases, however, GENP with random circulant pre-processing and with a single loop of iterative refinement has produced solution with desired accuracy, matching the output accuracy of GEPP. Furthermore GENP has performed similarly when it was applied to a nonsingular and well-conditioned input pre-processed with a Gaussian multiplier.

Table 9.1: Relative residual norms: GENP with Gaussian multipliers

dimension	**iterations**	**mean**	**max**	**min**	**std**
256	0	6.13×10^{-9}	3.39×10^{-6}	2.47×10^{-12}	1.15×10^{-7}
256	1	3.64×10^{-14}	4.32×10^{-12}	1.91×10^{-15}	2.17×10^{-13}
512	0	5.57×10^{-8}	1.44×10^{-5}	1.29×10^{-11}	7.59×10^{-7}
512	1	7.36×10^{-13}	1.92×10^{-10}	3.32×10^{-15}	1.07×10^{-11}
1024	0	2.58×10^{-7}	2.17×10^{-4}	4.66×10^{-11}	6.86×10^{-6}
1024	1	7.53×10^{-12}	7.31×10^{-9}	6.75×10^{-15}	2.31×10^{-10}
2048	0	4.14×10^{-9}	8.16×10^{-9}	8.27×10^{-10}	3.72×10^{-9}
2048	1	7.61×10^{-12}	1.08×10^{-11}	3.27×10^{-12}	3.89×10^{-12}
4096	0	5.02×10^{-7}	1.23×10^{-6}	1.14×10^{-7}	6.29×10^{-7}
4096	1	5.44×10^{-11}	1.53×10^{-10}	2.64×10^{-12}	8.52×10^{-11}

Table 9.2: Relative residual norms: GENP with Gaussian circulant multipliers

dimension	**iterations**	**mean**	**max**	**min**	**std**
256	0	8.97×10^{-11}	1.19×10^{-8}	6.23×10^{-13}	4.85×10^{-10}
256	1	2.88×10^{-14}	2.89×10^{-12}	1.89×10^{-15}	1.32×10^{-13}
512	0	4.12×10^{-10}	3.85×10^{-8}	2.37×10^{-12}	2.27×10^{-9}
512	1	5.24×10^{-14}	5.12×10^{-12}	2.95×10^{-15}	2.32×10^{-13}
1024	0	1.03×10^{-8}	5.80×10^{-6}	1.09×10^{-11}	1.93×10^{-7}
1024	1	1.46×10^{-13}	4.80×10^{-11}	6.94×10^{-15}	1.60×10^{-12}
2048	0	1.03×10^{-8}	2.87×10^{-8}	1.13×10^{-9}	1.59×10^{-8}
2048	1	3.74×10^{-13}	6.09×10^{-13}	9.69×10^{-14}	2.59×10^{-13}
4096	0	2.46×10^{-9}	4.17×10^{-8}	1.93×10^{-10}	2.05×10^{-9}
4096	1	7.82×10^{-13}	1.35×10^{-12}	2.02×10^{-13}	5.72×10^{-13}

Next we cover our tests of GENP pre-processed with some multipliers defined by means of combining matrices of Section 6. The test results are represented in Tables 9.5 and 9.6.

In this series of our tests we set $n = 128$ and applied GENP to matrices of (9.1) and six families of benchmark matrices from [BDHT13], pre-processed with multipliers combining the ones of following three basic families.

Family 1: The matrices APF of depth 3 (with $d = 3$) and with a (single) random permutation.

Table 9.3: Relative residual norms: GENP with circulant multipliers filled with ± 1

dimension	iterations	mean	max	min	std
256	0	2.37×10^{-12}	2.47×10^{-10}	9.41×10^{-14}	1.06×10^{-11}
256	1	2.88×10^{-14}	3.18×10^{-12}	1.83×10^{-15}	1.36×10^{-13}
512	0	7.42×10^{-12}	6.77×10^{-10}	3.35×10^{-13}	3.04×10^{-11}
512	1	5.22×10^{-14}	4.97×10^{-12}	3.19×10^{-15}	2.29×10^{-13}
1024	0	4.43×10^{-11}	1.31×10^{-8}	1.28×10^{-12}	4.36×10^{-10}
1024	1	1.37×10^{-13}	4.33×10^{-11}	6.67×10^{-15}	1.41×10^{-12}
2048	0	5.42×10^{-9}	1.59×10^{-8}	1.54×10^{-10}	9.04×10^{-9}
2048	1	1.17×10^{-13}	2.40×10^{-13}	5.14×10^{-14}	1.07×10^{-13}
4096	0	1.22×10^{-8}	2.47×10^{-8}	6.41×10^{-10}	1.21×10^{-8}
4096	1	2.29×10^{-13}	4.36×10^{-13}	1.05×10^{-13}	1.81×10^{-13}

Table 9.4: Relative residual norms: GENP for DFT(n) with Gaussian multipliers

dimension	iterations	mean	max	min	std
256	0	2.26×10^{-12}	4.23×10^{-11}	2.83×10^{-13}	4.92×10^{-12}
256	1	1.05×10^{-15}	1.26×10^{-15}	9.14×10^{-16}	6.76×10^{-17}
512	0	1.11×10^{-11}	6.23×10^{-10}	6.72×10^{-13}	6.22×10^{-11}
512	1	1.50×10^{-15}	1.69×10^{-15}	1.33×10^{-15}	6.82×10^{-17}
1024	0	7.57×10^{-10}	7.25×10^{-8}	1.89×10^{-12}	7.25×10^{-9}
1024	1	2.13×10^{-15}	2.29×10^{-15}	1.96×10^{-15}	7.15×10^{-17}
2048	0	2.11×10^{-11}	3.05×10^{-11}	1.64×10^{-11}	8.08×10^{-12}
2048	1	1.47×10^{-13}	2.73×10^{-13}	8.10×10^{-14}	1.09×10^{-13}
4096	0	1.36×10^{-10}	3.01×10^{-10}	4.52×10^{-11}	1.43×10^{-10}
4096	1	6.12×10^{-13}	9.69×10^{-13}	1.91×10^{-13}	3.93×10^{-13}

Family 2: Sparse circulant matrices $C = \Omega^{-1} D(\Omega \mathbf{v}) \Omega$, where the vector $\mathbf{v}$ has been filled with zeros, except for its ten coordinates filled with ± 1. Here and hereafter each sign $+$ or $-$ has been assigned with probability $1/2$.

Family 3: Sum of two inverse bidiagonal matrices. At first their main diagonals have been filled with the integer 101, and their first subdiagonals have been filled with ± 1. Then each matrix have been multiplied by a diagonal matrix $\mathrm{diag}(\pm 2^{b_i})$, where b_i were random integers uniformly chosen from 0 to 3.

We combined these three basic families of multipliers and tested GENP on their ten combinations, listed below. For each combination we have performed 1000 tests and have recorded the average relative error $||A\mathbf{x} - \mathbf{b}||/||\mathbf{b}||$ with matrices A from the seven benchmark families and vectors $\mathbf{b}$ being standard Gaussian vectors. Here are these ten combinations.

1. $F = I$, H is a matrix of Family 1.
2. $F = I$, H is a matrix of Family 3.
3. $F = H$ is a matrix of Family 1.
4. $F = H$ is a matrix of Family 3.
5. F is a matrix of Family 1, H is a matrix of Family 3.
6. $F = I$, H is the product of two matrices of Family 1.
7. $F = I$, H is the product of two matrices of Family 2.
8. $F = I$, H is the product of two matrices of Family 3.
9. $F = I$, H is the sum of two matrices of Families 1 and 3.
10. $F = I$, H is the sum of two matrices of Families 2 and 3.

We tested these multipliers for the same linear systems as in our previous tests in this section

and for six classes generated from Matlab, by following their complete description in Matlab and [BDHT13]. Here is the list of these seven test classes.

1. The matrices A of (9.1).
2. 'circul': circulant matrices whose first row is a standard Gaussian random vector.
3. 'condex': counter-examples to matrix condition number estimators.
4. 'fiedler': symmetric matrices generated with (i, j) and (j, i) elements equal to $c_i - c_j$ where $c_1, \dots, c_n$ are i.i.d. standard Gaussian variables.
5. 'orthog': orthogonal matrices with (i, j) elements $\sqrt{\frac{2}{n+1}} \sin \frac{ij\pi}{n+1}$.
6. 'randcorr': random $n \times n$ correlation matrices with random eigenvalues from a uniform distribution. (A correlation matrix is a symmetric positive semidefinite matrix with ones on the diagonal.)
7. 'toeppd': $n \times n$ symmetric, positive semi-definite (SPSD) Toeplitz matrices T equal to the sums of m rank-2 SPSD Toeplitz matrices. Specifically,

$$T = w(1) * T(\theta(1)) + ... + w(m) * T(\theta(m))$$

where $\theta(k)$ are i.i.d. Gaussian variables and $T(\theta(k)) = (cos(2\pi(i-j)\theta(k)))_{i,j=1}^{n}$.

In our tests, for some pairs of inputs and multipliers, GENP has produced no meaningful output. In such cases we filled the respective entries of Tables 9.5 and 9.6 with ∞.

GENP pre-processed with our multipliers of the 9th combination of three basic families, has produced accurate outputs without iterative refinement for all seven benchmark classes of input matrices. With the other combinations of the three basic families of our multipliers, this was achieved from 4 to 6 (out of 7) benchmark input classes. For comparison, the 2-sided pre-processing with PRBT-based multipliers of [BDHT13] and [BBBDD14] always required iterative refinement.

Table 9.5: Relative residual norms output by pre-processed GENP with no refinement iterations

class	1	2	3	4	5	6	7
1	2.61e-13	6.09e-15	∞	2.62e+02	7.35e-15	1.38e-12	3.04e-13
2	2.02e+02	4.34e-14	5.34e-16	∞	7.35e+02	5.27e-15	3.23e-15
3	4.34e-13	8.36e-15	∞	3.03e+02	1.94e-14	3.04e-13	3.21e-13
4	1.48e+01	1.36e-12	2.39e-16	1.01e-11	4.71e+01	5.09e-15	5.12e-15
5	3.71e-11	2.21e-14	∞	2.85e+01	5.83e-10	2.23e-12	1.34e-12
6	3.33e-13	9.36e-15	∞	3.66e-05	7.04e-15	3.75e-13	2.11e-13
7	7.76e-12	3.55e-14	9.91e+01	7.90e+00	7.75e+00	7.11e+00	1.05e+01
8	7.95e+00	9.55e-14	7.56e-16	∞	5.74e+03	6.51e-15	3.57e-15
9	5.36e-13	1.51e-14	4.26e-16	2.24e-11	3.68e-13	6.47e-15	4.92e-15
10	3.50e-12	8.43e-14	3.43e-13	2.90e-10	1.36e+01	3.53e-13	1.67e-13

Finally we present the results of our tests of GENP with additive pre-processing applied to the same $n \times n$ test matrices A of (9.1), but for $n = 128, 256, 512, 1024$. In this case we applied GENP to the matrix $C = A - UV^T$ where U and V were $n \times h$ random Gaussian subcirculant matrices, each defined by the n i.i.d. Gaussian entries of its first column and scaled so that $||A|| = 2||UV^T||$. Then we computed the solution $\mathbf{x}$ to the linear system $A\mathbf{x} = \mathbf{b}$ for a Gaussian vector $\mathbf{b}$ by substituting the SMW formula (8.6) into the equation $\mathbf{x} = A^{-1}\mathbf{b}$.

We present the test results in Table 9.7. The statistics were taken over 100 runs for each n. The results changed little when we scaled the matrices U and V to increase the ratio $||A||/||UV^T||$ to 10 and 100.

Table 9.6: Relative residual norms output by pre-processed GENP followed by a single refinement iteration

class	1	2	3	4	5	6	7
1	1.13e-15	6.90e-17	∞	1.12e+00	5.23e-17	2.10e-16	1.05e-15
2	5.07e-04	7.71e-17	1.03e-16	∞	4.40e+02	1.99e-16	1.19e-15
3	1.14e-15	7.34e-17	∞	5.43e-13	5.15e-17	2.24e-16	1.10e-15
4	1.55e-03	6.19e-17	1.31e-16	5.69e-13	2.69e+02	2.13e-16	1.17e-15
5	9.80e-16	6.96e-17	∞	6.75e+01	5.35e-17	2.47e-16	9.84e-16
6	1.08e-15	6.13e-17	∞	6.35e-13	5.08e-17	1.86e-16	1.05e-15
7	3.47e+01	6.17e-17	2.61e+06	5.21e+00	5.31e-17	1.97e-16	9.97e-16
8	2.56e-04	6.67e-17	1.15e-16	∞	7.96e+02	1.98e-16	1.08e-15
9	9.81e-16	7.44e-17	3.99e-17	6.40e-13	5.09e-17	2.02e-16	1.15e-15
10	9.79e-16	8.32e-17	1.14e-16	7.34e-13	4.07e+01	2.23e-16	1.04e-15

Table 9.7: Relative residual norms of GENP with Gaussian subcirculant additive pre-processing

n	h	iterations	mean	max	min	std
128	4	0	1.47e-10	2.51e-09	1.05e-12	3.85e-10
128	4	1	1.58e-14	3.39e-13	1.26e-15	4.48e-14
256	4	0	8.14e-10	2.87e-08	1.20e-11	3.02e-09
256	4	1	3.57e-14	1.16e-12	2.52e-15	1.24e-13
512	4	0	8.86e-09	3.03e-07	4.42e-11	3.44e-08
512	4	1	2.16e-13	1.35e-11	4.60e-15	1.36e-12
1024	4	0	2.12e-08	3.06e-07	1.45e-10	4.98e-08
1024	4	1	9.87e-14	1.95e-12	6.64e-15	2.38e-13

10 Conclusions

Gaussian elimination with partial pivoting is the workhorse for modern matrix computations, but it is significantly slowed down by communication intensive pivoting, both for inputs of small and large sizes. Pre-processing with random and fixed multipliers as well as by means of augmentation are efficient alternatives to pivoting according to the results of extensive tests in this paper and a number of previous papers. Our novel insight provides formal support for these empirical observations and embolden widening the search area for efficient pre-processors. We present our initial findings in our search for new classes of such pre-processors and confirm their efficiency with our numerical tests.

In [PZ16] and [PZa] our techniques yield similar results for the fundamental problem of *low-rank approximation of a matrix* and for the approximation of two singular spaces of a matrix associated with the two sets of its largest and smallest singular values separated by a gap, respectively.

For the latter subject our substantial new research progress is under way. We also plan to extend our present results to the computation of Rank Revealing LU Factorization of [P00].

Acknowledgements: We thank a reviewer for valuable comments and acknowledge support by NSF Grants CCF 1116736 and CCF–1563942 and PSC CUNY Award 67699-00 45.

References

[B85] J. R. Bunch, Stability of Methods for Solving Toeplitz Systems of Equations, *SIAM Journal on Scientific and Statistical Computing*, **6, 2**, 349–364, 1985.

[BBBDD14] M. Baboulin, D. Becker, G. Bosilca, A. Danalis, J. Dongarra, An Efficient Distributed Randomized Algorithm for Solving Large Dense Symmetric Indefinite Linear Systems, *Parallel Computing,* **40, 7***, 213–223, 2014.*

[BBD12] D. Becker, M. Baboulin, J. Dongarra, Reducing the Amount of Pivoting in Symmetric Indefinite Systems. Parallel Proc. Applied Math., PPAM 2011, Lecture Notes in Computer Science (LNCS), **7203**, 133–142, Springer, 2012.

[BCD14] G. Ballard, E. Carson, J. Demmel, M. Hoemmen, N. Knight, O. Schwartz, Communication Lower Bounds and Optimal Algorithms for Numerical Linear Algebra, *Acta Numerica* (an invited paper), **23**, 1–155, 2014.

[BDHT13] M.Baboulin, J.Dongarra, J.Herrmann, S.Tomov, Accelerating Linear System Solutions Using Randomization Techniques, *ACM Trans. Math. Software (TOMS)*, **39, 2**, 2013.

[BP94] D. Bini, V. Y. Pan, *Polynomial and Matrix Computations, Volume 1: Fundamental Algorithms*, Birkhäuser, Boston, 1994.

[BV88] W. Bruns, U. Vetter, *Determinantal Rings, Lect. Notes Math.,* **1327**, Springer, 1988.

[CD05] Z. Chen, J. J. Dongarra, Condition Numbers of Gaussian Random Matrices, *SIAM. J. on Matrix Analysis and Applications*, **27**, 603–620, 2005.

[DDF14] S. Donfack, J. Dongarra, M. Faverge, M. Gates, J. Kurzak, P. Luszczek, I. Yamazaki, A Survey of Recent Developments in Parallel Implementations of Gaussian Elimination, CONCURRENCY and COMP.: PRACTICE and EXPERIENCE, *Concurrency Comp.: Pract. Exper.* 2014, (wileyonlinelibrary.com), DOI: 10.1002/cpe.3306

[DE] *Dillon Engineering*, http://www.dilloneng.com/fft_ip/parallel-fft.

[DKM06] P. Drineas, R. Kannan, M.W. Mahoney, Fast Monte Carlo Algorithms for Matrices I–III, *SIAM J. on Computing*, **36, 1**, 132–206, 2006.

[DL78] R. A. Demillo, R. J. Lipton, A Probabilistic Remark on Algebraic Program Testing, *Information Processing Letters*, **7**, **4**, 193–195, 1978.

[DS01] K. R. Davidson, S. J. Szarek, Local Operator Theory, Random Matrices, and Banach Spaces, in *Handbook on the Geometry of Banach Spaces* (W. B. Johnson and J. Lindenstrauss editors), pages 317–368, North Holland, Amsterdam, 2001.

[EGH13] Y. Eidelman, I. Gohberg, I. Haimovici, *Separable Type Representations of Matrices and Fast Algorithms. Volume 1. Basics. Completion Problems. Multiplication and Inversion Algorithms. Volume 2. Eigenvalue method*, Birkhauser, 2013.

[ES05] A. Edelman, B. D. Sutton, Tails of Condition Number Distributions, *SIAM J. on Matrix Analysis and Applications*, **27**, **2**, 547–560, 2005.

[FKV98] A. Frieze, R. Kannan, S. Vempala, Fast Monte-Carlo Algorithms for Finding Low-rank Approximations, *J. of ACM*, **51**, 1025–1041, 2004. Proc. version in *39th FOCS*, 1998.

[G11] J.F. Grcar, Mathematicians of Gaussian Elimination, *Notices of the Amer. Math. Society*, **58, 6**, 782–792, June/July 2011.

[GL13] G. H. Golub, C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, 2013 (fourth addition).

[GS72] I. Gohberg, A. Sementsul, On the Inversion of Finite Toeplitz Matrices and Their Continuous Analogs, *Matematicheskiie Issledovaniia* (in Russian), **7**, **2**, 187–224, 1972.

[GTZ97] S. A. Goreinov, E. E. Tyrtyshnikov, N. L. Zamarashkin, A Theory of Pseudo-skeleton Approximations, *Linear Algebra and Its Applications (LAA)*, **261**, 1–21, 1997.

[GZT97] S. A. Goreinov, N. L. Zamarashkin, E. E. Tyrtyshnikov, Pseudo-skeleton Approximations by Matrices of Maximal Volume, *Mathematical Notes*, **62, 4**, 515–519, 1997.

[H02] N.J. Higham, *Accuracy and Stability in Numerical Analysis*, SIAM, Philadelphia, 2002.

[HMT11] N. Halko, P. G. Martinsson, J. A. Tropp, Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions, *SIAM Review*, **53, 2**, 217–288, 2011.

[KP91] E. Kaltofen, V. Y. Pan, Processor Efficient Parallel Solution of Linear Systems over an Abstract Field, *Proc. 3rd Ann. ACM Symp. on Parallel Algorithms and Architectures (SPAA'91)*, 180–191, ACM Press, New York, 1991.

[KS91] E. Kaltofen, B. D. Saunders, On Wiedemann's Method for Solving Sparse Linear Systems, *Proceedings of AAECC–5, LNCS*, **536**, 29–38, Springer, Berlin, 1991.

[M11] M. W. Mahoney, Randomized Algorithms for Matrices and Data, *Foundations and Trends in Machine Learning*, NOW Publishers, **3, 2**, 2011.

[P90] V. Y. Pan, On Computations with Dense Structured Matrices, *Math. of Computation*, **55, 191**, 179–190, 1990. Procs. version in ISSAC'89, 34–42, ACM Press, NY, 1989.

[P00] C.-T. Pan, On the existence and computation of rank-revealing LU factorizations, *Linear Algebra and its Applications (LAA)*, **316**, 199–222, 2000.

[P01] V. Y. Pan, *Structured Matrices and Polynomials: Unified Superfast Algorithms*, Birkhäuser/Springer, Boston/New York, 2001.

[P15] V. Y. Pan, Transformations of Matrix Structures Work Again, *LAA*, 465, 1–32, 2015.

[P16] V. Y. Pan, How Bad Are Vandermonde Matrices? *SIMAX*, **37**, **2**, 676–694, 2016.

[PGMQ] V. Y. Pan, D. Grady, B. Murphy, G. Qian, R. E. Rosholt, A. Ruslanov, Schur Aggregation for Linear Systems and Determinants, *Theoretical Computer Science*, *Special Issue on Symbolic–Numerical Algorithms* (D. A. Bini, V. Y. Pan, and J. Verschelde editors), **409**, **2**, 255–268, 2008.

[PIMR10] V. Y. Pan, D. Ivolgin, B. Murphy, R. E. Rosholt, Y. Tang, X. Yan, Additive Preconditioning for Matrix Computations, *LAA*, **432**, 1070–1089, 2010.

[PQ10] V. Y. Pan, G. Qian, Solving Homogeneous Linear Systems with Randomized Preprocessing, *Linear Algebra Appls. (LAA)*, **432**, 3272–3318, 2010.

[PQ12] V. Y. Pan, G. Qian, Solving Linear Systems of Equations with Randomization, Augmentation and Aggregation, *Lin. Algebra Appls. (LAA)*, **437**, 2851–1876, 2012.

[PQY15] V. Y. Pan, G. Qian, X. Yan, Random Multipliers Numerically Stabilize Gaussian and Block Gaussian Elimination: Proofs and an Extension to Low-rank Approximation, *Linear Algebra and Its Applications (LAA)*, **481**, 202–234, 2015.

[PQZ13] V. Y. Pan, G. Qian, A. Zheng, Randomized Preprocessing versus Pivoting, *Linear Algebra and Its Applications (LAA)*, **438, 4**, 1883–1899, 2013.

[PQZC] V. Y. Pan, G. Qian, A. Zheng, Z. Chen, Matrix Computations and Polynomial Root-finding with Preprocessing, *Linear Algebra and Its Applications (LAA)*, **434**, 854–879, 2011.

[PSZ15] V. Y. Pan, J. Svadlenka, L. Zhao, Estimating the Norms of Circulant and Toeplitz Random Matrices and Their Inverses, *Lin. Algebra Appls. (LAA)*, **468**, 197–210, 2015.

[PW08] V. Y. Pan, X. Wang, Degeneration of Integer Matrices Modulo an Integer, *Linear Algebra and Its Applications (LAA)*, **429**, 2113–2130, 2008.

[PZ15] V. Y. Pan, L. Zhao, Randomized Circulant and Gaussian Preprocessing, in *Procs. 17th Int. Workshop on Compute Algebra in Sci. Comp. (CASC'2015)*, (V.P. Gerdt, V. Koepf, and E.V. Vorozhtsov, editors), *LNCS Science*, **9301**, 359–373, Springer,

[PZ16] V. Y. Pan, L. Zhao, Low-rank Approximation of a Matrix: Novel Insights, New Progress, and Extensions, Proc. *11th Int.l Comp. Sci. Symp. in Russia (CSR'2016)* (A.Kulikov, G.Woeginger, eds.), St. Petersburg, 2016, *LNCS*, **9691**, 352–366, Springer,

[PZa] V. Y. Pan, L. Zhao, New Studies of Randomized Augmentation and Additive Preprocessing, *LAA*, 2017, http://dx.doi.org/10.1016/j.laa.2016.09.035. arxiv 1412.5864.

[SST06] A. Sankar, D. Spielman, S.-H. Teng, Smoothed Analysis of the Condition Numbers and Growth Factors of Matrices, *SIAM J. Matrix Anal. Applics.*, **28**, **2**, 446–476, 2006.

[T00] E. E. Tyrtyshnikov, Incomplete Cross Approximation in the Mosaic-skeleton Method, *Computing*, **64, 4**, 367–380, 2000.

[T11] J. A. Tropp, Improved analysis of the subsampled randomized Hadamard transform, *Adv. Adapt. Data Anal.*, **3, 1–2**, Special Issue "Sparse Representation of Data and Images," 115-126, 2011.

[VVM07] R. Vandebril, M. Van Barel, N. Mastronardi, *Matrix Computations and Semiseparable Matrices,* **1**, The Johns Hopkins University Press, Baltimore, MD, 2007.

[W14] D.P. Woodruff, Sketching as a Tool for Numerical Linear Algebra, *Foundations and Trends in Theoretical Computer Science*, **10, 1–2**,1–157, 2014.

[XXG12] J. Xia, Y. Xi, M. Gu, A Superfast Structured Solver for Toeplitz Linear Systems via Randomized Sampling, *SIAM J. Matrix Analysis Appls. (SIMAX)*, **33**, 837–858, 2012.

[YC97] M.C. Yeung, T.F. Chan, Probabilistic Analysis of Gaussian Elimination without Pivoting, *SIAM J. Matrix Analysis and Applications (SIMAX)*, **18, 2**, 499–517, 1997.